



1506
UNIVERSITÀ
DEGLI STUDI
DI URBINO
CARLO BO

DIPARTIMENTO DI SCIENZE PURE E APPLICATE (DISPEA)

CORSO DI DOTTORATO DI RICERCA IN:
RESEARCH METHODS IN SCIENCE AND TECHNOLOGY
CICLO XXXV

PERSPECTIVES ON COMPUTER SCIENCE EDUCATION

SSD: INF/01

Coordinatore: Chiar.mo Prof. Alessandro Bogliolo

Supervisore: Chiar.mo Prof. Alessandro Bogliolo

Dottorando : Francesco Paolo Maiorana

ANNO ACCADEMICO 2021/22



1506
UNIVERSITÀ
DEGLI STUDI
DI URBINO
CARLO BO

DEPARTMENT OF PURE AND APPLIED SCIENCES

Ph.D. PROGRAMME IN:

RESEARCH METHODS IN SCIENCE AND TECHNOLOGY
CYCLE XXXV

PERSPECTIVES ON COMPUTER SCIENCE EDUCATION

ACADEMIC DISCIPLINE: INF/01 Computer Science

Coordinator: Prof. Alessandro Bogliolo

Supervisor:: Prof. Alessandro Bogliolo

Ph.D. student : Francesco Paolo Maiorana

ACADEMIC YEAR 2021/22

Dedication

In order of appearance

To my mother

Whose example and courage gave me the strength to overcome all the difficulties.

To my father

For never giving up.

To be continued ...

Acknowledgments

A special thanks goes to Prof. Ralph Morelli¹: “Ralph was an enthusiast of life, an optimist, a person who found humor in any situation, and someone who inspired many others. He exuded a playful sense of ease, often instilling calmness into many situations. Many found a friend in him, appreciating his sincerity, curiosity, and engaging conversation.”. I am a proud member of the many inspired by him who found a friend in him.

A special thought and thanks go to the many researchers and educators in Italy and worldwide,

- my colleagues both at school and universities in Italy and USA,
- the working group companions both the ones during this thesis (ITiCSE, IEEE, CSTA) and the previous working groups,
- the coauthors of the many published and under review works, more that 40 during this thesis, more than 100 during my research practice,
- the reviewers of this thesis, and of my works for their insightful comments,
- the many reviewers of two special issues lead as guest editor,
- the co-reviewer with whom I collaborated, both as reviewer and associated co program chair, and shared opinions during the discussion phase for submitted work in journal and conferences,
- the many researchers and educators with whom, I collaborated during this thesis in organizing conferences (Scientix and CSTA) and panels (ITiCSE and Educon) at the Italian and international level, and the many who supported me in the first two decades of research experiences

listing them all will be a daunting task prone to forgetting someone.

Finally, at last but not least, a special thanks to my supervisor, Prof. Alessandro Bogliolo, who gave me the possibility to complete this Ph.D.

¹ <https://www.wakememorialpark.com/obituaries/Ralph-Morelli/#!/Obituary>

Perspectives on Computer Science Education

Index

<i>Dedication</i>	3
<i>Acknowledgments</i>	4
<i>Perspectives on Computer Science Education</i>	5
Index	5
List of works crafted during this thesis.	11
Introduction	14
Chapter 1: The state of the art on Computer Science Education research.....	20
1.1 Introduction	20
1.2. Computing tools and coffers.....	23
1.2.1. Physical computing tools	23
1.2.2. Soft computing tools	24
1.2.3. Unplugged tools.....	25
1.2.4. Coffers and resources	25
1.2.5. Lessons learned	26
1.3. Coaching.....	27
1.3.1. Coaching approaches: lessons learned and best practices.....	27
1.4. Contextualization	28
1.4.1 Enacted curriculum in England	29
1.4.2. Intended curriculum in Italy and the European Community	29
1.4.3 Intended curriculum in Nebraska, Kansas and the USA.....	29
1.4.3.1 Nebraska	30
1.4.3.2 Kansas.....	30
1.4.4 Intended curriculum guidelines	31
1.5. Curriculum	31
1.5.1 Enacted curriculum in England and the United Kingdom	31
1.5.2. Enacted curriculum in Italy and the European community.....	33
1.5.3 Enacted curriculum in Nebraska, Kansas and the USA	34
1.5.4 Lessons learned and best practices for implementing computing curriculum ...	35
1.6. Creativity.....	36
1.6.1 How does computing and CT promote creativity within STEAM?.....	36
1.7. Competitions	37
1.7.1 Competitions in the United Kingdom.....	37
1.7.2 Competitions in Italy and the European community	38
1.7.3 Experience report: the Google4CS@CT case	39

1.7.3 Competitions in Nebraska, Kansas and the USA	40
1.7.4.1. Nebraska	40
1.7.4.2. Kansas	40
1.7.4. Lessons learned and best practices.....	41
1.8. Checking.....	41
1.8.1 Lessons learned	42
1.9. Convenience	43
1.9.1 Experience report: The Master lab for in-service and pre-service and student teachers for special ability certification	44
1.9.2 How does computing and CT promote convenience and access for all learners within STEAM?.....	45
1.10. Challenges	45
1.11. Collaboration and Communication	47
1.12. Customs.....	48
1.13. Communities of Practices (CoP)	49
13.1 The interplay between small and large CoP: the case of Google4CS@CT	50
13.2 Lessons learned	50
1.14. Citizenship	51
1.15. Conclusions	51
Chapter 2: International comparison of intended and enacted curricula.....	53
2.1 Objectives and research questions	54
2.2 Methods	55
2.3 The country reports	55
2.4 The teacher survey	62
2.4.1 Intended Curriculum Observations.....	67
2.4.2 Enacted Curriculum Observations	68
2.5 Discussion and lessons learned	73
2.5.1 Pilot Study Curriculum Observations	74
2.5.2 Lessons Learned.....	75
2.5.3 Recommendations for Future Use.....	77
2.5 Teachers' Computer Science self-esteem.....	77
2.6 International comparison of Intended and enacted curricula.....	79
Chapter 3: Competencies.....	83
3.1 From CS2013 to CC2020.....	83
3.2 Level of Granularity	84
3.3 Common Format	85
3.4 Validation Methodology.....	85
3.4.1 Competency Statement Self-Evaluation.....	86
3.4.2 Competency Statement Peer-Validation.	87

3.5. Preliminary Analysis.....	89
3.6. Discussion	90
3.6.1 Steps for Re-expressing LO-based Curricula.....	90
3.7. Examples of competencies statements	91
3.8 What was Learned During the Process?	92
Chapter 4: Learning resource development and assessment	96
4.1 International comparison of enacted computing curriculum	100
4.1.1 Database and Project management.....	100
4.1.1.1 Lesson learnt	107
4.1.2 Algorithms, Data, Networks, Security and Ethics	109
4.1.2.1 Programs, Algorithms, Data and CT.....	109
4.1.2.2 Networking, Internet and Security	111
4.1.2.3 Ethics.....	112
4.1.2.4 Mapping to standards.....	113
4.1.2.5 Comparison and Lessons Learned.....	113
4.1.2.6 Lessons learned.....	114
4.1.3 Computer Systems and Human Computer Interaction.....	116
Rationale of exploring Computer Systems and Human Computer Interface	116
Computer Systems	117
Human Computer Interaction	120
Mapping to standards	123
Lesson learned	123
4.2 A Curriculum and a Learning Path for a First Course on Computing	127
4.2.1 Design, development and evaluation of a first course on computing.....	127
Design principles	127
The learning trajectories.....	128
Evaluation of results and discussion	136
4.3.2 Learning trajectories	138
The data collection process	138
Learning trajectories for a first course in computing	139
Function first	139
Lesson learned	144
Loop first.....	144
Learning trajectory evaluation, lessons learned and best practices.....	146
Variable first.....	146
Learning trajectory evaluation, lessons learned and best practices.....	147
Comparison of the learning trajectories.....	148
4.3 A learning path for computing and civic education	149

4.3.1 Outreach activities related to Coding, Computational and Algorithm thinking.	151
4.3.2 Competencies in Computer and Information Literacy, Civic and Citizenship Education, and Computer Science	152
4.3.3 Learning path	153
4.3.4 Discussion	154
4.3.5. Importance of social science theories in education	155
Approach	155
Experience reports.....	159
Guidelines.....	161
4.4 A second computing course on algorithms and data structures.....	165
4.4.1 How to address online learning.....	165
4.4.2 How to address students not majoring in Computing	166
4.4.3 Course content.....	167
4.4.4 Pedagogies	168
4.4.5 Technologies	169
4.4.6 Outcomes	169
4.4.7 Attitude survey.....	174
4.5 The recursive module: content, technologies, and learning trajectories.....	176
4.5.1 The content	178
4.5.2 Case study	179
4.5.3 Animation design.....	180
4.5.4 Design of the formative assessment tool	180
4.5.5 Preliminary field test	181
4.5.5.1 Finding for the field test with Undergraduate students	181
4.5.5.2 Finding for the field test with high school students	182
4.6 Curriculum on Foundation of programming for talented youth	184
4.6.1 Foundation of programming: syllabus	184
4.6.1.1 W1: Design and implement an imperative and procedural program	184
4.6.1.2 W2: Recursion, Data structures and Object Oriented Programming	185
4.6.1.3 W3: Design tools: UML diagrams, E/R, EER, and event Programming ...	187
Chapter 5: Accessibility in the curriculum	189
5.1 Literature review on technologies, pedagogies and content	189
5.2 The Italian master experience	190
5.3 The Alabama experience	191
5.5 Discussion	192
Chapter 6: Community of Practices	193
6.1 Scientix Teachers Ambassadors for Critical Digital Literacy and Active Citizenship	194
6.1.1 Process and products.....	196

6.1.2 Scientix ambassadors on the field experiences	198
6.1.3 Designing learning resources for digital citizenship, data, and information literacy	199
Experience evaluation and impact	202
Conclusions	202
6.2 The Scientix CoP	204
6.1.1 A class site as a tool for reading, reflecting, writing, coding, community building, and leadership development	204
6.1.2 Inclusive computing for digital humanities	205
6.1.3 A curriculum for digital citizenship, data, and information literacy	207
6.1.4 Discussion	208
6.3 Pre-service Teachers Formation via Systems Thinking and TPCK from Italy, England, and USA	208
6.3.1 Introduction	209
6.2.1 Relationship between Systems Thinking and TPCK	215
6.2.2 Candidate Selection	216
6.2.3 Technological Pedagogical Content Knowledge (TPCK) Framework	216
6.2.4 Internship and Practicum	217
6.2.5 Transiting to In-Service	220
6.2.6 Lessons Learned	221
6.2.7 Summary	229
6.4 Professional work	231
6.3.1 ITiCSE 25th anniversary committee	232
6.3.2 Convenor at the special Interest Group in Educational Technologies	233
6.3.3 American Educational Research Association	234
6.3.4 Computer Science Teachers Associations committees	235
6.3.5 3rd Italian Scientix conference organizing committee	235
6.3.6 Computing at School	235
6.3.7 Guest editor for the Informatics in Education Journal	236
6.3.8 Special issue proposal for the IEEE Transaction on Education	237
6.5 Interlink of research and on the field educational practice	238
Chapter 7: Data related to Computer Science education	240
7.1 Topic Indexes	240
7.2 User reliability indexes	241
7.3 Resource Indexes	242
7.4 Use of attention metadata	243
8. Conclusions	244
8.1 Computing education: from content, pedagogies, and technologies, to cognitive, psychological, and philosophical aspects	245

List of works crafted during this thesis.

- Cassel, L., Gal-Ezer, J., & Maiorana, F. (2020). Celebrating 25 ITiCSE Conferences: Involving Everyone to Look at the Past to Construct the Future. *ACM SIGCSE Bulletin*, 52(3), 6–7.
- Clear, A., Clear, T., Vichare, A., Charles, T., Frezza, S., Gutica, M., Lunt, B., Maiorana, F., Pears, A., Pitt, F., Reidsel, C., & Szykiewicz, J. (2020). Developing Competency Statements for Computer Science Curricula: The Way Ahead. *WG10. Proceedings of 25th ACM International Conference on Innovation and Technology in Computer Science Education*. ACM.
- Clear, A., Clear, T., Vichare, A., Charles, T., Frezza, S., Gutica, M., Lunt, B., Maiorana, F., Pears, A., Pitt, F., Riedesel, C., & Szykiewicz, J. (2020). Designing Computer Science Competency Statements: A Process and Curriculum Model for the 21st Century. *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education*, 211–246.
- Cristaldi, G., Csizmadia, A., Richards, G., Quille, K., Riedesel, C., & Maiorana, F. (2022). The intervention, intersection and impact of social sciences theories upon computing education. *Proceedings of the EDUCON2022 – IEEE Global Engineering Education Conference*.
- Cristaldi, G., Richards, G., Csizmadia, A., Riedesel, C., & Maiorana, F. (2020). Special education for special students: Experience reports from teachers' educators and in the field teachers. *Proceedings of the 13th Annual International Conference of Education, Research and Innovation*. <https://library.iated.org/view/CRISTALDI2020SPE>
- Maiorana F., Cristaldi G. (under review). *Design of a learning journey. VI Seminar 'INVALSI data: A tool for teaching and scientific research'*.
- Maiorana F., Gras-Velazquez A., Paladino E., Cristaldi G., Niewint-Gori J., Mishra S., Trifas M., Sharmin S., Gonzalez C. (2023). *IEEE Transactions on Education (ToE)*, Special Issue on "Coding, Computational, Algorithmic, Design, Creative, and Critical Thinking in K-16 Education". Retrieved April 2023 at <https://iee-edusociety.org/toe-special-issue-coding-computational-algorithmic-design-creative-and-critical-thinking-k-16>
- Maiorana, F., Altieri, S., A., C., M., L., L., N., M., P., A., S., & G, C. (2023). Scientix Teachers Ambassadors for Critical Digital Literacy and Active Citizenship. *Italian Journal of Education Technology*.
- Maiorana, F., & Csizmadia, A. (2022). Introduction to the Special Issue: Educational Alliance: A Global Partnership between Education and Industries Addressing the 4th Sustainable Development Goal. *Informatics in Education*, 21(4), 569–570.
- Maiorana, F., Richards, G., & Csizmadia, A. (2023). *Pre-service Teachers Formation via Systems Thinking and TPCK from Italy, England, and USA*. Studying Teacher Education.
- Falkner, K., Sentance, S., Vivian, R., Barksdale, S., Busuttil, L., Cole, E., Liebe, C., Maiorana, F., McGill, M., & Quille, K. (2019a). An International Comparison of K-12 Computer Science Education Intended and Enacted Curricula. *Proceedings of the 19th Koli Calling International Conference on Computing Education Research*, 1–10.

- Falkner, K., Sentance, S., Vivian, R., Barksdale, S., Busuttil, L., Cole, E., Liebe, C., Maiorana, F., McGill, M., & Quille, K. (2019b). An International Study Piloting the MEasuring Teacher Enacted Computing Curriculum (METRECC) Instrument. *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education*, 111–142.
- G., C., Csizmadia, A., Riedesel, C., & Maiorana, F. (2020). Inclusive computing for digital Humanities. *Accepted Abstract at the ATEE Spring Conference 2020. Conference Postponed To*.
- Maiorana, F. (2019). Interdisciplinary Computing for STE (A) M: a low Floor high ceiling curriculum. *Innovations, Technologies and Research in Education*, 37. <https://doi.org/10.22364/atee.2019.itre>
- Maiorana, F. (2021). From High School to Higher Education: Learning Trajectory for an Inclusive and Accessible Curriculum for Teachers and Their Students. *International Journal of Smart Education and Urban Society (IJSEUS)*, 12(4), 36–51.
- Maiorana, F. (2022a). Dati una risorsa preziosa. In *Consapevoli in rete. Percorsi per la cittadinanza digitale*. Pearson.
- Maiorana, F. (2022b). Impatto ambientale della rete e degli archivi dati. In *Consapevoli in rete. Percorsi per la cittadinanza digitale*. Pearson.
- Maiorana, F. (2022c). Verificare le informazioni e le fonti. In *Consapevoli in rete. Percorsi per la cittadinanza digitale*. Pearson.
- Maiorana, F. (2023a). Impatto ambientale della rete. In *Il mondo che vorrei. Sostenibilità: Istruzioni per l'uso*. Sanoma, Paravia, Edizioni Scolastiche Bruno Mondadori, linx. <https://sanomaitalia.it/mystem#volumedigitale>
- Maiorana, F. (2023b). Risorse educative aperte" in Le STEM nella Rete Scientix Italia: Riflessioni sulle STEM nella didattica. *Aprile*.
- Maiorana, F. (2020). A flipped design of learning resources for a course on algorithms and data structures. *Proceedings of the 9th International Conference on Interactive Collaborative and Blended Learning*.
- Maiorana, F., Altieri, S., Colli, A., Labbri, M., Nazzaro, L., Porta, M., Severi, A., & Guida, M. (2020). Scientix Teacher Ambassadors: A passionate and creative professional community linking research and practice. *Proceedings of the 13th Annual International Conference of Education, Research and Innovation*. <https://library.iated.org/view/MAIORANA2020SCI>
- Maiorana, F., & Cristaldi, G. (2022). *From Data to Coding and responsible digital citizenship: The*
- Maiorana, F., & Cristaldi, G. (2023). From Data to Coding and responsible digital citizenship: The design of a learning journey. In *Surveys on students: The INVALSI national and international tests. VI Seminar 'INVALSI data: A tool for teaching and scientific research'*.
- Maiorana, F., Csizmadia, A. R., G., & Riedesel, C. (2020). Recursion and iteration: A combined approach for algorithm comprehension. *Proceedings of the 9th International Conference on Interactive Collaborative and Blended Learning (ICBL)*.

- Maiorana, F., Csizmadia, A., & Richards, G. (2020a). Attention metadata in learning resource repositories. *Proceedings of the 13th Annual International Conference of Education, Research and Innovation*. <https://library.iated.org/view/MAIORANA2020ATT>
- Maiorana, F., Csizmadia, A., & Richards, G. (2020b). Managing Data and Projects: Lessons learnt from comparing computing curricula. *Proceedings of the 9th IEEE International Conference on Engineering, Technology and Education*.
- Maiorana, F., Csizmadia, A., & Richards, G. (2020c). P12 Computing in Italy, England and Alabama, USA. *Proceedings of the 21st Annual Conference on IT Education (SIGITE)*.
- Maiorana, F., Csizmadia, A., & Richards, G. (2021). Metrics Based on Attention Metadata for Learning Resource and Assessment Repository. *Proceedings of the 12th IEEE Global Engineering Education Conference (EDUCON)*.
- Maiorana, F., Nazzaro, L., Severi, A., Colli, A., Porta, M., G., C., & Labri, M. (2022). Reflections on inclusive leadership education: From professional communities of practices to students. *IUL Research*, 3, 5,.
- Maiorana, F., Quille, K., Csizmadia, A., Richards, G., & G, C. (2022). From Coding to Algorithms: Learning path for computing and civic education competencies development. *EDUCON2022 – IEEE Global Engineering Education Conference*.
- Niewint-Gori, J., Goracci, S., Minichini, C., Cosssu, C., Lonigro M, Maiorana F. (2023a). 3° conferenza Scientix Italia 2022 book of abstract. *INDIRE*. https://www.indire.it/wp-content/uploads/2022/12/BoA_Scientix_Italia_2022_R_JNG.pdf
- Niewint-Gori, J., Cosssu, C., Lonigro M, Maiorana F. (2023b). Le STEM nella Rete Scientix Italia: Raccolta di Esperienze Scuola dell'Infanzia, Scuola Primaria. *Aprile*. https://www.indire.it/wp-content/uploads/2023/04/Indire_Scientix_Esperienze_STEMIntegrate_InfanziaPrimaria_17.pdf
- Niewint-Gori, J., Cosssu, C., Lonigro M, Maiorana F. (2023c). Le STEM nella Rete Scientix Italia: Raccolta di Esperienze Scuola secondaria I grado. *Aprile*. https://www.indire.it/wp-content/uploads/2023/04/Indire_Scientix_Esperienze_STEMIntegrate_SecI Grado_17.pdf
- Niewint-Gori, J., Cosssu, C., Lonigro M, Maiorana F. (2023d). Le STEM nella Rete Scientix Italia: Raccolta di Esperienze Scuola secondaria II grado. *Aprile*. https://www.indire.it/wp-content/uploads/2023/04/Indire_Scientix_Esperienze_STEMIntegrate_SecII Grado_17.pdf
- Niewint-Gori, J., Cosssu, C., Lonigro M, Maiorana F. (2023e). Le STEM nella Rete Scientix Italia: Riflessioni sulle STEM nella didattica. *Publicato online sul sito di Indire - Scientix Italia, Aprile 2023 ISBN: 979-12-80706-43-0 Copyright © INDIRE 2023*.
- Quille, K., McGill, M., Vivian, R., Falkner, K., Sentance, S., Barksdale, S., Busuttill, L., Cole, E., Liebe, C., & Maiorana, F. (2020, June). An International Pilot Study of K-12 Teachers' Computer Science Self-Esteem. *Proceedings of 25th ACM International Conference on Innovation and Technology in Computer Science Education*. ACM.

Introduction

In recent years an important intellectual movement around the world has focused attention on the importance of introducing Informatics as a basic literacy competence from primary school to tertiary education and throughout the whole learning path. This has been coupled by restating the central role of education in improving the quality of life of all people around the world. Education is considered the most powerful means to achieve the basic rights of people from basic and physiological needs such as food, water, warmth, rest, up to self-actualization through achieving personal full potential including creative activities (OECD, 2021) and self-fulfillment (Maslow & Lewis, 1987), (McLeod, 2007). The fourth United Nations (UN) sustainable development goal (United Nations, 2015) focuses on quality education and aims at 'ensuring inclusive and equitable quality education and promoting lifelong learning opportunities for all'. Actions have been taken to achieve the UN educational development goal with global crisis undermining the goals, not the effort employed. According to the 2022 UN Report (United Nations, 2022), "cascading and interlinked crises are putting the 2030 Agenda for Sustainable Development in grave danger, along with humanity's very own survival". The needs for incisive action for all sustainable goals, including education, has become of paramount importance, one requiring collective effort.

In the same direction for quality education for all goes the strong movement for quality computing education that, in a decade at the school level, has led to national laws changing the program of study, e.g. the National Curriculum in England: computing programs of study (DFE, 2013a) up to designing a computing framework across Europe (Caspersen et al., 2022b), (Caspersen et al., 2022a), with similar efforts in higher education. This effort has impacted the formal educational system and has been strongly supported by the non-formal educational system with initiatives across the globe starting from the code.org movement (Code.org et al., 2022) and the EU Code Week initiatives (Sirocchi et al., 2022).

The importance of computing and digital competencies is remarked in the recommendation of large international organizations such as the Organization for Economic Co-operation and Development (OECD) which has recently released their council recommendations on creating better opportunities for young people². In the document the council recommend actions along 5 major directions with a particular emphasis on involving young people aged 15-29 years:

- 1) Acquire relevant knowledge and develop appropriate skills and competencies.
- 2) Support in transition into and within the labor market, improving labor market outcomes.
- 3) Promote social inclusion and youth well-being beyond economic outcomes.
- 4) Establish the legal, institutional, and administrative settings to strengthen the trust of young people in government and their relationship with public institutions.

² <https://legalinstruments.oecd.org/en/instruments/OECD-LEGAL-0474>

- 5) Reinforce administrative and technical capacities to deliver youth-responsive service and address age-based inequalities through close collaboration across all levels of governments.

At the national level, actions targeting the 4th and 5th point are addressed by the National Resilience Program inside the European Union next generation plan.

Recently researchers around the world have proposed a systemic approach (Fuller & Kim, 2022; Sengeh, 2022) as a way to achieve a holistic development of students (Datnow, 2022). Research (Maiorana & Cristaldi, 2023) has highlighted the importance of a system thinking approach to transform schools (Fuller & Kim, 2022), supporting teachers, the mind and heart of the educational system around the world, through adequate policies supporting their continuing professional learning (Boeskens et al., 2020a; OECD, 2019b; OECD, 2021b), advocating for upskilling and investing in people through bottom-up solutions and insights (OECD/OPSI, 2020).

It is clear that a global perspective on teaching is needed for an ample and systemic reform of the educational system involving all the actors of the educational community and in this respect a wise use of digital technologies can contribute towards a positive digital and green transition of society. According to the European Parliament's decision to establish the Digital Decade Policy Programme 2030, "Digital technologies should contribute to achieving broader societal outcomes that are not limited to the digital sphere but have positive effects on the everyday lives and well-being of citizens. If it is to be successful, the digital transformation should go hand-in-hand with improvements as regards democracy, good governance, social inclusion and more efficient public services".

In this regard, this work aims to make a small contribution in this direction by providing, as a proof of concepts, both the process and the products, i.e., some exemplar curricula and learning resources that embrace a global perspective on teaching through the lens of system thinking both computing and the characters of the whole learning community as a means to fulfill a holistic development.

In this work we will:

- 1) Offer examples of the process supported by research practices for crafting computing learning resources to support the acquisition of "knowledge and develop appropriate skills and competencies" as evinced by international survey and competencies analyses elaborated from the comparison of the ACM 2013 and 2020 computing curriculum.
- 2) Provide case studies, in accordance with the Technological Pedagogical Content Knowledge (TPKC) framework, related to the design, development and assessment of inclusive and accessible computing learning resources coupled with pedagogical best practices and supported by cutting edges technologies. The proposed learning resources are suited to a national computing course for all high school students and undergraduate courses for students not majoring in computer science, aimed at

impacting the learning path of the high school students both at the national and international level.

- 3) Offer research approaches for structuring collaborations inside communities of practices guiding the research and educational daily practices.
- 4) Present a data collection and analysis framework aimed at guiding and supporting the educational practices of the educators and the student educational practices and offer to the educational community, including the parents, early detection of learning difficulties.

In particular the work and the reflections have been guided by the underlying questions that has inspired, under the hood, the first 30 years and more of teaching practices and the more than a decade of research work on computing education of the author, namely, according to (Datnow et al., 2022a), (Datnow et al., 2022b):

“The first question is rhetorical, aiming to engage both heart and mind in considering efforts to build and rebuild academically focused education systems into humanistic education systems that also support the social, emotional, moral, and civic development of students.

- 1) What would it mean—and what would it take—to build education systems that develop every child as would that child’s own parents?”

The second question is empirical, aiming to draw a diverse global audience into productive, evidence-informed conversation about complex and contentious issues of collective interest, one central issue being potential synergies between the pursuits of academic and holistic student development.

- 2) Is there evidence that it is even possible to (re)build academically focused education systems to support holistic student development?

This effort has been central for education from an ancient time and for holistic student development it is essential to pursue character education (Watts & Kristjánsson, 2022) starting from school (Arthur & Kristjánsson, 2022). Ancient lessons from philosopher and thinkers has to be brought to new life to contribute to the systematic development of virtues aiming at developing human flourishing, i.e. well being, since good character is both conducive and constructive of overall wellbeing (Watts & Kristjánsson, 2022).

The long term goal is through a System Thinking approach contribute to an holistic learner development including the character and virtue for the wellbeing of the person, the community and the planet.

To contribute with a grain of sand to these ambitious goals the following chapters are presented

- 1) Chapter one will offer an overview of the state of research in a broad spectrum of topics related to computer science education, namely: Computing tools and coffers, i.e. toolboxes, focusing on tools and learning resources, considered valuable, for computing education; Coaching as blending of pedagogical approaches; Contextualization, i.e. an overview of intended curricula in each country; Curriculum

with a focus on educator-created curricula; Creativity and how computing and CT promote creativity; Competitions such as Bebras and ICPC and CCSC programming contests; Checking and Computing and Computational Thinking assessment; Convenience and how computing promotes special abilities; Challenges faced by learners and educators, and ways to overcome those challenges; Collaboration and Communications - how computing promotes collaboration and communications; Customs and ethical principles laying the foundation of the educational process with a particular emphasis on the computing domain; Communities of Practice (CoP) of educators and how they represent one of the most powerful vehicles for sharpening the competencies and skills of whole learning communities by bringing together educational practice and research; Citizenship with an emphasis on digital citizenship and wisdom on how CT and computing can be taught, learned, and applied for the social good. This overview offers a robust and solid support for the successive research work providing an epistemological and narrative review with a comprehensive set of scientific study guiding the conceptualization, development, and assessment of the successive work.

- 2) Chapter two will focus on an international comparison of intended curricula (Falkner, 2019b), (Porter, 2001), i.e. enforced or suggested by national guidelines at the state level, and the teacher enacted curricula, i.e., actual curricular content taught by teachers that students engage with in the classroom, and educational practices around the world. This is accomplished by designing, assessing a survey instrument, and conducting the data analysis. The instrument represents a research tool to acquire a broad view of the research areas highlighted in the previous chapter. The data analysis represents the foundation for the curricula design process presented in chapter 4, highlighting the differences between intended and enacted curricula and teachers needs related to computing education. The self-esteem section offers support for designing, conducting and assessing research experiences in professional communities of practices presented in chapter six. The developed instrument, scientifically assessed and validated represent the first example in this direction
- 3) Chapter Three, leveraging the ACM Computing Curricula 2020 and its reflections on a framework of competency-based educational principles closely aligned with other skills and qualifications frameworks presents. The work demonstrates one way in which the transition from current learning-outcomes-based practices to the competency-based practices can be approached. And, the same time, the paper provides reflections on computing competencies that must guide the curricula design, development, and assessment process. In conjunction with the instruments previously presented the competency based model offer a scientific support for the

disci, multi, inter and transdisciplinary computing curricula development, the core of the thesis work.

- 4) Chapter Four aims to reach the first and second goals presented above. On the basis of the previous discussion, it presents the design, development and in some cases assessment of several computing curricula along national and international experience, both in a working context at Kansas State University and in research experiences spurred from international working groups and professional communities of practices carried out during this doctorate. The reflections are supported by more than thirty years of teaching experience in high school, undergraduate and graduate course as well as outreach educational activities in computing. By presenting both the processes and the products the chapter is intended to advocate and offer a proof of concept for:
- a. the importance and impact of research on the educational practice
 - b. the importance and relevance of computer science education as a for itself research domain
 - c. the importance of integrating computing in all curricula at all levels of K-16 education from kindergarten to higher and adult education
 - d. provide educational resources for students training and educator training and professional development both pre and in service.

The interlinking of civic education not restricted to the digital domain, computing and overall education find a comprehensive view in the framework for character education (Arthur & Kristjánsson, 2022) composed of the following building blocks:

- a) Intellectual virtues for discernment, right action and the pursuit of knowledge, truth and understanding. Examples in this direction can be found in the proposed educational resources for navigating information and data across the web.
- b) Character traits enabling us to act well in situation requiring an ethical response. An example in the realm of respect of the world around us can be found in the proposed education resources on sustainable development.
- c) Character traits necessary for engaged responsible citizenship, contributing to the common good, e.g., service and volunteering in civic society for being a change agent and per the contribution on learning resource for teachers and their students.
- d) Character traits that have an instrumental value in enabling the intellectual, moral and civic virtues, e.g. confidence, determination, motivation, perseverance, resilience, leadership and teamwork, all quality essential for quality education in all domain.

- 5) Chapter Five extends the previous chapter by dealing with accessibility in education from the perspective of designing, developing, and assessing accessible resources and pedagogies and technologies supporting the use of the learning resources. According to the universal design for learning a careful planning and design of educational resources with a variety of content, pedagogical approaches, type of languages, media and technologies, and different and increasing levels of difficulties for in-context real life activities is beneficial for the whole learning community.
- 6) Chapter Six highlights the importance of research practices for daily educational activities and how an active participation in professional communities of practices can support this process. As a proof of concept, direct experiences and several case studies will be presented.
- 7) Finally, chapter seven will focus on data related to Computer Science education with an emphasis on designing a data collection mechanism aimed at improving educational practice, e.g., for early detection and remediation of low educational engagement. The proposed data collection process and the suggested analysis support the usage and assessment of the learning resources presented in Chapter 4.

Chapter 1: The state of the art on Computer Science

Education research

Objective of the chapter: Provide an epistemological and comprehensive narrative review as emerging from the comparison of educational experiences of researchers around the world, i.e., Italy & Europe, Nebraska & USA, England & United Kingdom

Approach: starting from a systematic review conducted according to the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) guidelines a narrative and epistemological review is distilled arising from international discussions and comparisons

Result achieved and novelty: an annotated bibliography to a large literature on computational thinking. It cites nearly 200 papers.

Significance for the state of the art and the narrative of the work: it provides a solid research foundation framing the successive steps.

1.1 Introduction

The primary issue for this chapter is in what resources exist and how they can contribute to academic enrichment practices for fostering development of Computing and Computational (CT), algorithmic (AT), design (DT), creative (Cre) and critical (Cri) thinking in students so that they can excel in (hopefully) all STEAM education. More fundamentally, can educators be empowered with best practices and materials so as to impart the above thinking to their students, regardless of the subject area? For example, students who only learn how to follow instructions (as in a program), may become proficient in following instructions, but probably not gain proficiency in solving the underlying problems independently. In such a restricted environment, they gain only the capabilities of a (non-thinking) computer, not of the computer programmer, or more generally of the creators of all STEAM solutions. Is this what students are looking for or do they prefer to look for other ways to express their creativity? Can this be considered quality education?

One of UNESCO's sustainability goals is quality education for all (UNESCO, 2017), (Owens, 2017). The paramount importance of education is clearly stated by researchers (Howells, 2018a). Quality education allows for "an integrated approach" with mutual sustainment involving different activities pursuing different goals. This international effort for quality education is comprehensive, incorporating the dissemination of computing knowledge to all citizens, involving all educators, using all levels of education, extending into all types of educational systems, and considering all stages of life, have all been indicated to be pursued from early development (Cutts et al., 2018) and has been the focus of the Computer Science education community during its entire existence.

Some notable examples around the world in this direction are the following:

1. Computing At School (CAS)³ which was, is and will be the driving force of the National Curriculum in England and Scotland.
2. Computer Science Teachers Association (CSTA) which released a framework (CSTA, 2016) and a Standard, both for students (CSTA, 2017d) and teachers (CSTA, 2020b), covering Computer Science (CS) education from kindergarten through 12th grade, which drives the effort in the USA.
3. Code.org⁴ (Franke & Osborne, 2015) with an incisive action to have everyone learn CS, working with educational departments across the USA and around the world with initiatives covering Europe⁵ (Sirocchi et al., 2022) and Italy (Corradini & Nardelli, 2021), and the United Kingdom.
4. CSforAll (Santo et al., 2018), a movement dedicated to bringing high quality Computer Science to all school students in order to prepare them for college.
5. Informatics for All (M. E. Caspersen et al., 2018a) coalitions supported by the ACM Europe Council, Informatics Europe, and the Council of European Professional Informatics Societies (CEPIS) aims at introducing Informatics as a fundamental discipline for all learners.
5. The Consortium for Computing Sciences in Colleges (CCSC)⁶ promotes CS in two and four years Colleges and Universities in the USA and represents the glue between K-12 school and higher education.
6. Additional leading educational organizations like OECD (Howells, 2018b), (OECD learning, 2030) and ACM, and more specifically SIGCSE.

A focus on competencies is driving the effort in new curricula development and comparison on 21st-century competencies, and skills (Binkley et al., n.d.), such as Computational Thinking (CT), play a key role (Wing, 2020) despite a debate (Denning, 2017) lasting more than 80 years (Tedre & Denning, n.d.). It is their (and our) position that CT should be included with an interdisciplinary approach in all disciplines and should involve all stakeholders. According to the various operational definitions of CT it is possible to argue that:

- CT can be interpreted as a transversal and transversal set of skills that can be used as a means to acquire and to develop broad competencies like the one proposed in (Bocconi, Chiocciariello, Dettori, Ferrari, & Engelhardt, 2016).

³ <https://www.computingatschool.org.uk/>

⁴ <https://code.org/>

⁵ <https://codeweek.eu/>

⁶ <https://www.ccsc.org/about-us/history/>

- “more tools in the mental toolbox seems like a worthy goal” (Tedre & Denning, n.d.).

Despite the remarkable effort cited above and despite the excellent results obtained thus far (according to objective data analysis (Guzdial, 2020a)) “we have not yet created popular computing education. We reach very few students”. Sustained by these efforts, the authors under the umbrella of CT and in the light of their experiences covering three nations and two continents and spanning decades up to an entire life will present:

- 1) **Computing tools and coffers** focusing on the technological side of the multifaceted educators’ competencies. We will provide an overview from accessible and configurable microcomputers such as the BBC Micro:bit, and Raspberry Pi microcomputer to computing supported by block based languages and unplugged activities and discuss how these tools promote CT within STEAM. Readily available coffers and resources will be briefly presented.
- 2) **Coaching** as to what blending of pedagogical approaches best fits the class and the individual needs of each learner.
- 3) **Contextualization**, i.e. an overview of intended curricula in each country, and how CT can serve as an interdisciplinary glue among STEAM courses.
- 4) **Curriculum**, for example, educator-created curricula in each country, and how CT can and is used by educators to support and sharpen their competencies in Technologies, Content and Pedagogies. We will briefly state how to assess CT resources.
- 5) **Creativity** and how computing and CT promote creativity within STEAM.
- 6) **Competitions** such as Bebras and ICPC and CCSC programming contests and other competitions such as those organized by teachers for students in a community of practice supported by a Google for Education grant, and how these competitions sharpen CT skills and STEAM education. We will discuss how competitions can be used for CT formative and summative assessments.
- 7) **Checking**, after a discussion on CT and its multiple facets, we will discuss CT assessment and how it can contribute to engagement in STEAM education.
- 8) **Convenience** and how computing and CT promote convenience and special abilities within STEAM.
- 9) **Challenges** faced by learners and educators, and ways to overcome those challenges.

- 10) **Collaboration and Communications** - how computing and CT promote collaboration and communications within STEAM.
- 11) **Customs** and ethical principles laying the foundation of the educational process with a particular emphasis on the computing domain.
- 12) **Communities of Practice (CoP)** of educators and how they represent one of the most powerful vehicles for sharpening the competencies and skills of educators by bringing together educational practice and research. The authors will report on experiences nurtured by participating in CoP or leading CoP from a local to a nation-wide level. These experience reports will highlight how CT was used in various CoPs to enhance content, pedagogies, collaboration, communications creativity and convenience competencies.
- 13) **Citizenship** with an emphasis on digital citizenship and wisdom on how CT and computing can be taught, learned, and applied for the social good.

The above topics will be organized in sections. Each section will have a brief overview and state of the art regarding the topic, a sub-section for each country, namely England, Italy and the USA, with a particular emphasis on Nebraska and Kansas when appropriate, and a final section with lessons learned and best practices. Concluding remarks and further work will be presented in the last section.

1.2. Computing tools and coffers

Nowadays many computing tools can be used and are useful for teaching and learning in interdisciplinary settings. We will classify these computing tools along three main categories: physical, software and unplugged. For a list of tools used to develop CT in STEAM, the reader can reference (Falkner et al., n.d.), for which teachers around the world have been surveyed for the tools and resources they use in their daily practices. As stated in (Repenning et al., n.d.), CT Tools must be designed, developed, and used to minimize the cognitive overhead during the coding phase by supporting users through three fundamental stages of the CT development cycle: problem formulation, solution expression, and solution execution/evaluation. Beside this, technology represents a way for innovation in education (Iskander, 2018).

1.2.1. Physical computing tools

These types of computing tools facilitate the interface with the real world, for example by collecting data, or controlling and connecting devices. They can be broadly classified as:

- 1) Tiny low-cost desktop computers like Raspberry Pi⁷ provide learners with a real computer at an affordable price which needs only a mouse, keyboard, and monitor in order to have a fully featured computer equipped with an operating software, Integrated development environment and sophisticated mathematical software; in short, all the tools necessary to develop and sharpen CT and 21st century competencies.
- 2) Microcontrollers such as the BBC Microbit⁸ and Arduino⁹. These allow interfacing all types of external devices, controlling them and receiving inputs from many sensors, thus representing an optimal intermediary to integrating systems, collecting data from external environments, and building interconnected smart devices. Open source (Pearce, 2012)¹⁰ resources allow us to replicate all the artifacts from software to hardware.
- 3) All other computers, e.g. desktop and laptop

Other tools that can be used to foster CT in an interdisciplinary setting include drones, robots, and 3D printers. Movements like the Makers (Rode et al., 2015) represent an informal gathering of educators devoted to using physical tools in their education activities.

1.2.2. Soft computing tools

We categorized the software tools useful to build CT into STEAM into the following classifications:

- 1) Integrated Development Environments (IDE) that support programming activities.
These environments can be subdivided into:
 - a. Block based languages that enable coding by visually snapping together blocks. A host of products are now available, from App Inventor which provides mobile programming and interfacing with microcontrollers like the above mentioned, to Snap and its dialects which support programming many kinds of devices ranging from robots (Xia & Zhong, 2018) to 3D printers¹¹ (C. Johnson & Bui, 2015), (Koschitz & Rosenbaum, 2012) with applications ranging to edge computing applications like databases (Gorman et al., 2014) and ontologies (Ceriani & Bottoni, 2017) to humanities and Latin (Zhou et al., 2016).

⁷ <https://www.raspberrypi.org/>

⁸ <https://www.microbit.org/>

⁹ <https://www.arduino.cc/>

¹⁰ <https://www.oshwa.org/>

¹¹ <http://beetleblocks.com/>

- b. Hybrid languages that allow bidirectional switching between block and text, or text style entry of blocks (Monig et al., 2015) or a combination of text style editing for expression-level details with drag and-drop blocks as in (Kölling et al., 2019), (Kölling, 2018). The results obtained in using hybrid-based languages in an educational setting are promising (Deng et al., 2020). Considerations of the impact of block and hybrid based languages on the learnability of programming, an aspect of CT development can be found in (Bau et al., 2017).
- c. Text based languages, with a set of more than 3,000 educational possibilities.

2) Apps and even games (Code.org, 2019) can be software tools that can be used to teach and learn CT concepts, either by actually playing the games themselves or using the tools themselves to develop games. Reviews and guidelines in this direction can be found in (C. Johnson et al., 2016) and (McGill et al., 2018).

Movements like CoderDojo (Bocconi, Chiocciariello, Dettori, Ferrari, Engelhardt, et al., 2016) can play an important social role, gathering school learners of all ages from primary to high school, their parents and their teachers.

1.2.3. Unplugged tools

There are many tools that can be used for unplugged activities. The simplest one is pencil and paper, but the set is so vast that it is not feasible to list them all here. The CS Unplugged Book (Bell et al., 2015), (Bell et al., 2012a) includes a rich set of activities that use a varied set of tools. The unplugged activities can take many forms, involving learners in all stages of their learning path from primary school through secondary education and beyond with puzzle-based learning (Levitin & Levitin, 2011). The positive effects of unplugged activities and the development of CT has been demonstrated in (C. K. Looi et al., 2018).

1.2.4. Coffer and resources

Among the resources that can be used for developing CT, we want to mention Open Educational Resources (OERs) and Open Data. Openness allows an easier distribution, sharing and reusing, repurposing and localization of learning resources (C.-K. Looi et al., 2014). Communities of Practice like Scientix (Billon et al., 2019) offer a free localization of learning resources in any language.

At the same time, Open Data allows for multiple perspectives on data gained from the studies of different groups of people. Studies involving open data can range through all domains from science (Molloy, 2011) to the humanities (Coddington, 2015).

1.2.5. Lessons learned

In order to address learner diversity based on social conditions, accessibility to technology, and learning styles, learning resources and pedagogical approaches should offer the opportunity for learners to choose their preferred tools and domains of applications. For a review of the state of the art on tools supporting CT the reader can reference (Hsu et al., 2018).

The authors make the following observations:

- 1) Media rich content enhances the range of options for computing tools that could be used to achieve a set of competencies and take advantage of overlapping domains covered by such tools. Examples in this direction are programmable hardware devices that can be coded both using low level machine language, near the device, and more abstractly through a high-level programming language, which can be either visual or text based. In line with (Tsarava et al., 2019) an unplugged, puzzle based learning, coding and making approach should be pursued.
- 2) Use both block-based and text-based programming languages to provide a progression pathway from primary to higher education (D. D. Garcia et al., 2012a), (D. Garcia et al., 2015), (Bau et al., 2017).
- 3) Best efforts have to be put in place to lower the digital divide. Content has to be designed for fast access even in the presence of slow internet connections. Content should be distributed freely to disadvantaged students. An IDE allowing both a cloud based and off-line usage should be preferred. Most of the cloud-based IDEs are open source and allow downloading all the code needed for executing it off-line with a Web browser.
- 4) Use commenting and documenting tools as means to sharpen communication competencies. Commenting and documenting may often be the best debugging tools.
- 5) Use privileged collaborating environments and tools (Griffin & Care, 2014) that require collaboration among participants to solve the task at hand.
- 6) Try to establish collaboration between open source and proprietary publishing houses: They have unique features allowing for better diffusion. In some countries the national level capillary distribution network of publishing houses is able to reach every school and every teacher, and still represent a means of diffusion complementing the on-line option. One way to accomplish this is to involve in international projects learning resource developers and publishing houses with a business plan that allows free distribution over the Internet and paid printed resources.

We advocate a multifaceted approach to education, offering when possible multiple options of tools, resources and approaches. For example, a student who might not feel comfortable with the plugs and wiring required by devices may nevertheless be an eager designer using a software tool.

1.3. Coaching

Most modern pedagogies promote a student centered approach in which an active role is assigned to students. Contributing Student pedagogies, where students are encouraged to contribute to the learning of others and value their input, in the realm of computing have been proposed in (Hamer et al., 2008). One of the pedagogies that is advocated is an inquiry-based approach (Hazelkorn, 2015a), (European Commission et al., 2007) implemented in multiple flavors such as:

- 1) Peer Instruction¹² (L. Porter et al., 2016a), (B. Simon & Cutts, 2012)
- 2) Process Oriented Guided Inquiry Learning (POGIL)¹³ ('Education ambivalence', 2010)
- 3) Challenge Based Learning (C. Johnson et al., 2009).

The inquiry-based approach makes use of a flipped classroom strategy (J. Bishop & Verleger, 2013) in which students have access to resources before class. Employing a blend of pedagogical approaches provides more opportunities to find a match with an individual student's preferred learning strategies. Promising results are presented in (Chiquito et al., 2020) where resources, using a flipped classroom approach, are presented to students with linked activities employed to assess student knowledge before class begins. Errors detected in these responses are used as starting points for a class discussion in which students, using a peer instruction approach, are requested to solve problems. In (Sharples et al., 2016) the authors present ten innovations they believe will have the potential to induce a major shift in educational practice from teaching to learning and assessment. Those practices will impact the sharpening of CT competencies and will benefit from use of computing tools.

1.3.1. Coaching approaches: lessons learned and best practices

The main lessons learned on pedagogical approaches and best teaching practices can be summarized as follows:

- 1) Apply a flipped approach in learning resource design and development. Initiate the learning resource with a formative assessment test, then discuss the results of the

¹² www.peerinstruction4cs.org/

¹³ <http://guidedinquiry.org/>

assessment in the next session. This approach is consistent with the productive failure strategy presented in (Sharples et al., 2016).

- 2) Build bridges spanning successive class levels through a mentorship program. Students from higher level classes mentor students from lower classes. For the initial and final class sessions, use orienting activities to assist transitioning students, e.g. middle school students entering high school, and exiting high school students entering colleges and universities. This can be implemented with the teach-back approach described in (Sharples et al., 2016).
- 3) Strive to prepare designers by emphasizing design and abstraction activities. Sustain all students with the help of group activities, taking care to mix into each group both “designers” and “implementers”. Allow students the freedom to choose their project designs and implementations for the learning resources, while giving appropriate credit (Borges et al., 2017). This will enhance a sense of ownership for the work developed. Always encourage them, as this can yield extraordinary results (Hug et al., 2013). This can be framed in the design-thinking pedagogical approach described in (Sharples et al., 2016).
- 4) Be inclusive - involve the whole class in the process. Having students with different abilities in the class is a great resource for the entire class, as long as they are involved in sustaining these students in solving the real problems they encounter.
- 5) Allow peer teaching. Usefulness on a large scale of this approach is described in (Kundisch et al., 2012). The pedagogical approach is known as “learning from the crowd”.

1.4. Contextualization

A sustained worldwide effort has resulted in a revision of the mandatory state level computing curricula, such as the Computing Curriculum in England (DFE, 2013a), the Australian Curriculum (A.C.A.R.A., 2016), and the New Zealand Technology Curriculum (T.K.I. Ministry of Education, 2017). The main impact of these curricula is a shift from a focus on how to use technologies and tools in which the student is viewed as a consumer, to a focus on building artifacts, both in software and hardware, using technologies and tools with which the student is a creator. In the USA this effort is summarized in the joint CSTA and ISTE *Standards for Computer Science (CS) Teachers* which complements “The Universal Outcomes for Students” found in the K-12 Computer Science Framework and the CSTA Computer Science Standards (CSTA, 2017a), (CSTA, 2020a), (CSTA, 2017b). Their plan is to develop supporting material for using the standard by 2020. The effort has resulted in a steady progression in the number of schools offering computing classes and the number of states recognizing credits towards graduation with computing classes (Code.org, 2019), (Codeorg, 2018). In Europe the report by (M. E. Caspersen et al., 2018b), after reviewing the Informatics

situation in Europe for high schools (Vahrenhold et al., 2017)¹⁴ and higher education¹⁵, defines a two tier strategy to introduce computing and CT in all schools as an independent subject for specialization courses and as interdisciplinary integration with other domains. Many more initiatives exist such as in Israel, Lithuania (Benaya et al., 2017), Finland and northern Europe (Bocconi et al., 2018) and Singapore (Seow et al., 2019). In the following we briefly review the intended curricula in England, Italy, and two states in the USA: Nebraska and Kansas.

1.4.1 Enacted curriculum in England

The English Computing Programme of Study (DFE, 2013a) was introduced into the English National Curriculum in 2013 for compulsory teaching in both state funded primary and secondary schools. It replaced the previous Information Communications Technology (ICT) component in the compulsory national curriculum (Gove, 2012), (Royal Society, 2012) (Royal Society, 2017). Central to the Computing Programme of Study are the two interwoven threads of creativity and CT, as indicated in the study's opening sentence:

“A high-quality computing education equips pupils to use computational thinking and creativity to understand and change the world.” (DFE, 2013a).

1.4.2. Intended curriculum in Italy and the European Community

In Italy a significant effort over several years has resulted in national level laws and guidelines that state the importance in introducing CT starting at the primary level and proceeding to lower secondary school, i.e. middle school, with a proposal of a formal verification at the end of the first school cycle. The 107/2015 Italian law has produced a national plan for a digital school¹⁶ to develop those computing competencies across all school levels and across all disciplines. A proposal for a national computing curriculum put forward by a national consortium of universities has been described in (CINI. Consorzio Interuniversitario nazionale per l'informatica., 2017) and (Forlizzi et al., 2018a).

1.4.3 Intended curriculum in Nebraska, Kansas and the USA

In the USA, each state has or is putting forward state level guidelines to be implemented by school districts in each country. We briefly review the state level guidelines in Nebraska and Kansas.

¹⁴ <http://cece-map.informatics-europe.org/>

¹⁵ <https://www.informatics-europe.org/data/higher-education/>

¹⁶ https://www.istruzione.it/scuola_digitale/index.shtml

1.4.3.1 Nebraska

According to the Nebraska Department of Education¹⁷ there is an ongoing strong effort with “key stakeholders currently developing the Computer Science Education State Plan. The plan articulates the goals for computer science, strategies for accomplishing the goals, and timelines for carrying out the strategies”. Here is a summary of the Computational Thinking Standards and the Programming Standards¹⁸:

Computational Thinking:

- Create algorithms, or series of ordered steps, to solve problems (starting in K, master in grades 5-12).
- Decompose a problem into smaller more manageable parts (starting in 1, master in grades 5-12).
- Collect, analyze, and represent data effectively (starting in 2, master in 7-12).
- Demonstrate an understanding of how information is represented, stored, and processed by a computer (starting in 3, master in 8-12).
- Optimize an algorithm for execution by a computer (starting in 8, master in 10-12).
- Create simulations/models to understand natural phenomena and test hypotheses (starting in 6-7, master in 11-12).
- Evaluate algorithms by their efficiency, correctness, and clarity (starting in 8-9, master in 12).

Programming Standards:

- Write programs using visual (block-based) programming languages (Scratch, code.org), (starting in 1, master in 4-12).
- Create and modify animations, and present work to others (starting in 2, master in 4-12).
- Write programs using text-based programming languages (starting in 6, master in 11-12).
- Create web pages with a practical, personal, and/or societal purpose (starting in 6, master in 11-12).

1.4.3.2 Kansas

The Kansas Department of Education (KSDE) has put forward the Computer Science Standards Grades P-12 (KSDE. Kansas state department of education., 2019). The standard delineates learning objectives aimed at providing to all students, especially in underrepresented populations, rigorous fundamental Computer Science concepts, and a pathway to progress. For high school students, the standards differentiate between two

¹⁷ <https://www.education.ne.gov/nce/cis/>

¹⁸ <https://cdn.education.ne.gov/wp-content/uploads/2018/04/NEK12Tech.pdf>

classes of standards: L1 intended for all students, and L2 intended for CS career oriented students. The standard encourages schools to offer additional L2 courses for these students. The standard is primarily based on the K-12 Computer Science Framework and the subsequent Computer Science Teachers Associations standard.

KSDE still does not call for computing as a standalone subject in K-8, but does for 9-12. The implementation guidelines call for an integration starting from the lower grades. Particular attention is made to support rural districts that do not have the resources to implement it as a subject.

1.4.4 Intended curriculum guidelines

We recommend the following:

- 1) Extend the national and state level guidelines to undergraduate computing for non-majors. In some nations there is at least one course in each undergraduate degree. Framing this richness with a national level standard and national guidelines will be a benefit for the whole community.
- 2) Involve all stakeholders in the national committees and working groups. Limiting participation to universities will deprive the commissions and the working groups of the voices of their teachers and educators.

1.5. Curriculum

The richness of educator-enacted curriculum as emerged from (Falkner, Sentance, Vivian, et al., 2019a), (Falkner, Sentance, & Vivian, 2019), (Falkner, Sentance, Vivian, et al., 2019c), (Quille et al., 2020) is so extensive that it is impractical to summarize here. In Europe, reports such as (Licht et al., 2017a) provide an interesting overview of the best results in Europe. Here we just cover some examples in each country trying to follow a learning trajectory guided by age, i.e., primary, middle, high school, college, and higher education.

1.5.1 Enacted curriculum in England and the United Kingdom

In order for both primary and secondary teachers to comprehend what computational thinking is all about, Computing At School (CAS), the subject association for computer science teachers in the United Kingdom, established a Computational Thinking Working Group. This working group of computing subject experts were commissioned to investigate computational thinking and the implications for embedding and teaching computational thinking in the classroom. One of the tangible outputs of this group was the production of *Computational Thinking: A Guide for Teachers* (Csizmadia, Curzon, & Dorling, 2015).

In England, a number of national innovative initiatives and projects have been established to promote computational thinking within STEAM subjects for learners, such as Barefoot

Computing Project, Bebras Computing Challenge, and Digital Schoolhouse; and engineering education initiatives which promote computational thinking, such as BLOODHOUND Model Car Challenge and LEGO First League.

The Barefoot Computing Project (Berry et al., 2015)¹⁹ was initially funded by the English Department of Education to promote computational thinking concepts and approaches to inservice primary teachers as part of a national orchestrated continuous professional development programme which are summarized in Figure 1.



Figure 1 – Computational Thinking Concepts and Approaches

Subsequently, the project has been and continues to be funded by BT, a telecommunications organization, as part of its corporate social responsibility programme and commitment to develop learner’s digital skills.

Initially, the project commissioned a group of primary computing subject experts to develop a set of accessible cross-curriculum resources in order to embed computational thinking across the primary curriculum including STEAM subjects. At the present time, 68 resources have been produced.

The following table indicates how some of the 68 Barefoot Computing resources promote computational thinking concepts and approaches within STEAM subjects.

Computational Thinking Concepts	Barefoot Computing Resource	Computational Thinking Attitudes
Patterns, Abstraction	Data Dash	Collaborating
Logic	Bug in the Water Cycle	Debugging

¹⁹ <https://www.barefootcomputing.org/>

Abstraction, Decomposition, Algorithms, Evaluation	Classroom Monitor	Sound	Creating
Abstraction, Algorithms	Fossil Animation	Formation	Collaborating

Table 1 – How Barefoot Computing resources promote Computational Thinking Concepts and Activities

The initial resources have subsequently been refreshed and the relationship between computational thinking and different stages in the programming cycle, for example design, code and debugging, clearly articulated. This relationship is clearly illustrated in Figure 2.



Figure 2 – Computational Thinking and Programming

Resources were promoted and continue to be promoted to primary teachers in their own primary schools through professional development workshops delivered by Barefoot volunteers and Barefoot Ambassadors, and freely available to download via the Barefoot Computing website.

1.5.2. Enacted curriculum in Italy and the European community

The work for primary and middle school was pioneered by initiatives including “Code in your Classroom, Now” (Bogliolo, 2016), (De Rosa et al., 2017) which involved a community of

more than 30,000 teachers²⁰. A high school enacted curriculum was developed and utilized for teachers' professional development and pre-service teacher preparation and undergraduate computing courses for non-majors, (Maiorana, 2019a), (Maiorana, 2021c) resulting from a multiyear teaching and research experience (Giordano & Maiorana, 2014b), (Giordano & Maiorana, 2015a). The proposed curriculum aligned with the Technological, Pedagogical Content Knowledge model (De Rossi & Trevisan, 2018a), (Maiorana et al., 2017a), (Maiorana, Richards, Lucarelli, Miles, et al., 2019) a rich set of technologies and tools, pedagogical approach and content. In particular, different learning trajectories have been presented and discussed with an indication on the best technologies and pedagogies suited for each learning trajectory. One of these learning trajectories focuses on a function first approach, in order to present the role of functions and their usage in recursion as early as possible. Reviews of research (McCauley et al., 2015), (Rinderknecht, 2014) have highlighted effective research-based strategies to include recursion using different contents, contexts, practices, analogies, and tools. Initial studies have highlighted that, by adopting multiple approaches, non-major computing students will have a greater understanding of recursive sorting algorithms. Further studies in the adoption of this strategy are worth pursuing.

For both undergraduate and graduate masters courses for non-majors the richness of an enacted curriculum has led to most universities determining the content of their courses. It is common for the majority of university and college degree programmes to have at least one computing course in their programme of study. Unification in this direction should be sought.

1.5.3 Enacted curriculum in Nebraska, Kansas and the USA

In Nebraska there are many examples of curricula and initiatives. Among these it is possible to recall:

- 1) A curriculum, developed by one of the authors, to introduce computers and problem-solving with computers for applications in the sciences and engineering. It includes problem analysis and specification, algorithms, programming in a high-level language, and data representation and processing.
- 2) The EngageCSEdu²¹ projects involving researchers to develop peer reviewed and assessed educational resources (Craig & EngageCSEdu, 2020) designed according to engagement practices (Monge et al., 2015).

In Kansas, an effort in which one of the authors is co-involved as designer and developer, has been undertaken to develop the Computational Core certificate²². This is an online

²⁰ <https://blog.scientix.eu/2020/03/coding-recent-history-in-italy/>

²¹ <https://engage-csedu.org/>

²² <https://www.cs.ksu.edu/core/>

curriculum useful for the school and undergraduate community at a level far beyond the local school and university community.

1.5.4 Lessons learned and best practices for implementing computing curriculum

From the above discussion and the authors' experience, the following recommendations for the elected content developer can be highlighted:

- 1) Offer a rich set of communication means. This implies:
 - a) a rich set of computing tools used from hardware to software and unplugged,
 - b) A rich set of learning paths, leaving the freedom to teachers, educators and their students, to choose the one that best fits the context at hand and to the individual learner needs.
 - c) A rich set of content and assessment activities to fulfill as much as possible learner educational needs, still leaving the freedom for students to either choose or propose the activities they like the most.
- 2) Adopt a student-centered approach promoting a “flipped book” as described previously.
- 3) Implement short self-contained assessment activities, also incremental activities that build and expand on previous experiences. An incremental approach should be sought while preserving independence of each activity.
- 4) Avoid covering certain learning paths such as the functional approach at the end of the curriculum. An approach directing students to search iterative and recursive solutions should be fostered as early as possible. This consideration arises from:
 - a) Enabling concepts needs time and repetitive practice to be fully mastered.
 - b) Avoiding the burden of unlearning one way of thinking to learn a new one.

Regarding the last best outlined practices, the authors are concerned about applying CT too narrowly, connecting it closely to computing concepts in only procedural paradigms that focus on step by step processes in solving problems. There are other effective paradigms that are employed successfully in other disciplines ranging from mathematics to the fine arts. For example, the von Neumann architecture model of a computer has unfortunately dominated our thinking about what computing means.

The functional programming paradigm (and the related declarative paradigm) grew out of mathematics. It does not employ strict sequencing of operations, though there may be dependencies resulting from arguments passed into functions. Devising algorithms to solve or transform mathematical equations may be difficult, whereas relying on inference engines may reduce cognitive overload.

1.6. Creativity

There exist various definitions of creativity:

- 1) According to the Cambridge dictionary²³: “the ability to produce or use original and unusual ideas”
- 2) According to the Merriam-Webster dictionary²⁴: “the ability to create”
- 3) According to the Treccani definitions²⁵, the intellectual process divergent from the normal abstract logical process.
- 4) According to J.P. Guilford (Guilford, 1967), it is characterized by: particular sensitivity to problems, ability to produce ideas, flexibility of principles, originality in ideation, ability to synthesize, ability to analyze, ability to define and structure one's experiences and knowledge in a new way, breadth of the field of ideation, ability to evaluate.
- 5) According to (Romero et al., 2017) is “a context-related process in which a solution is individually or collaboratively developed and considered as original, valuable, and useful by a reference group”.

From the above definitions it is clear that creativity is a high order cognitive competency that is required in STEAM, and it permeates all disciplines and all aspects of our life. Among the projects linking CT and creativity, it is possible to recall (L. D. Miller et al., 2013). Where the authors demonstrate that the addition of creative exercises to computing courses for Computing, STEM and Humanities majors students, not only improves CT knowledge and skills, but by combining hands-on problem solving, guided analysis and reflection, real-world CS applications allows students to leverage their creative thinking skills to “unlock” their understanding of CT. The same path of linking of CT and creativity and other soft skills is advocated in (Lemay & Basnet, 2017) even if the authors did not find a link between CT skill development and academic performance. In (Pecanin et al., 2019) the authors report that by applying business process methodologies coupled with a constructive and multidisciplinary approach they found an increase in innovation and the student ability to develop innovative ideas.

1.6.1 How does computing and CT promote creativity within STEAM?

On the basis of the teaching and educational experience of the authors, the following educational practices are suggested:

1. Foster and require students to look for, find and share at least two solutions of each given problem. Besides developing creativity, the approach fosters critical thinking and decision making, requiring the students to compare strengths and weaknesses of each solution and choose the best one based on objective and subjective

²³ <https://dictionary.cambridge.org/dictionary/english/creativity>

²⁴ <https://www.merriam-webster.com/dictionary/creativity>

²⁵ <http://treccani.it/enciclopedia/joy-paul-guilford/>

parameters (Giordano & Maiorana, 2013b). The same process is suggested for the teaching practice by presenting as many examples as possible (Savage & Csizmadia, 2018).

2. Foster moments for presentations and classroom reflection. Use web-based tools to promote discussion beyond the classroom time (Giordano & Maiorana, 2013f). Besides developing critical thinking, these activities facilitate metacognitive strategies, sharing with students not only the results of this approach but also the “basic functioning” of a meta-level analysis (Hoppe & Werneburg, 2019).
3. Build a collection of the best solutions and share these solutions with students. Reading solutions performed by others fosters creative and critical thinking. Allowing reuse and repurposing of resources greatly helps in producing better solutions while offering a richer set of services (Maiorana, 2018).
4. Emphasizes process over product (Giordano & Maiorana, 2013c), (Savage & Csizmadia, 2018).

1.7. Competitions

Introduction: the spirit of competition. Some notable examples: Bebras tasks International Olympiad of Informatics, and Consortium for Computing Sciences in Colleges programming contest.

1.7.1 Competitions in the United Kingdom

Within the United Kingdom a number of organizations run both national and international competitions to promote not only CT but also, more specifically, CT and related computational activities with a STEAM focus. Examples of such competitions include:

UK Bebras Computational Thinking Challenge²⁶: This is an annual competition, administered by the University of Oxford and supported by the Raspberry Pi Foundation, to both primary and secondary learners to solve a series of increasing more challenging online algorithmic puzzles within 40 minutes.

Astro Pi²⁷: The European Astro Pi Challenge is an European Space Agency (ESA) Education project in collaboration with the Raspberry Foundation which has a competitive element Mission Space Lab and a non-competitive element Mission Zero. With Mission Space Lab, learners work as investigators either individually or in small groups to design and conduct scientific investigations in space by designing, developing and deploying computer programs

²⁶ <http://www.bebas.uk/>

²⁷ <https://astro-pi.org/>

that will run on a Raspberry Pi microcomputer on board the International Space Station (ISS). The investigators will then have the opportunity to analyse and evaluate the data collected and report on the results of their scientific investigation.

do your :bit²⁸: This is an international challenge competition in which learners combine creativity and technology in solutions for Global Goals. This challenge has been created by micro:bit Foundation, Arm and World's Largest Lesson in partnership with UNICEF. To date, three challenges have been created: Introduction to Global Goals, Life Below Water and Life on Land. In these challenges learners learn about Global Goals, understand some of the problems that oceans and the natural world are encountering, and by using both computational thinking and creativity to generate ideas and solutions using appropriate technologies.

1.7.2 Competitions in Italy and the European community

There are many examples of competitions in Italy.

Bebras tasks (ALaDDIn): The Italian Laboratory for Promoting and Empowering Computing Pedagogies²⁹ organizes the Italian edition of the Bebras Challenge (Belletini & Palazzolo, 2020), (Belletini et al., 2019), (Dagiene & Futschek, 2008), (Dagienė & Stupurienė, 2016), (Hubwieser & Mühling, n.d.), (Izu et al., 2017) along with other activities ranging from algorithms development through a process that starts unplugged and ends with coding, to teacher training, workshops for schools, and radio broadcast aiming at “diffusing and promoting enthusiasm for Informatics.

Olympiad of Problem Solving: National competitions that, according to the aims posted in the competition web site³⁰: spreading CT through engaging activities relevant to all disciplines. It aims at empowering students' problem solving competences, spreading CT as a strategy to solve problems, methods to obtain a solution and universal language to communicate with others.

Italian Olympiad of Informatics: A competition³¹ that begins with a school level competition and proceeds, by increasing the breadth and the scope of computing concepts covered, through competitions at the regional, national, and international level, covering all the

²⁸ <https://microbit.org/do-your-bit/>

²⁹ <https://aladdin.di.unimi.it/index.html>

³⁰ <https://www.olimpiadiproblemsolving.it/web/index.php>

³¹ <https://www.olimpiadi-informatica.it/>

algorithm design techniques and data structures to solve practical problems. During the competitions students have to design and code a working solution to problem presented to them.

International STEAM awards: This competition³² aims at collecting ideas, projects and innovative solutions to global problems related to smart cities and sustainable communities, life and health, space and new boundaries, energy and environments. Students' projects are showcased and awarded.

1.7.3 Experience report: the Google4CS@CT case

The Google4CS@CT was an initiative funded by Google for Education with the aims of creating a local community of practice (CoP) organized around the idea of using CT in teacher educational practices and to create and share lesson plans. The University of Catania obtained the grant after an international competition with a proposal emerging from an international collaboration involving the Mobile Computer Science Principles Project (R. A. Morelli et al., 2014). The CoP designed and developed a series of seven teachers' workshops and two series of seven face to face lessons for students, run in cooperation with teachers who attended the previous workshop. The teachers' workshop was designed to provide the opportunity to explore together how to introduce computing activities into existing STEM curricula, and to include computing in daily education activities.

In the students' meetings, high school students with the aid of their teachers were involved in mastering the curriculum developed in the teacher workshop and applying it to design and develop a software project, e.g. an educational game (Dolgopolovas et al., 2018), or a simulation (Deshpande & Huang, 2011), (Magana & Jong, 2018) showcased in a local competition at the University. Mixing unplugged, tinkering, making, and remixing pedagogical approaches (Kotsopoulos et al., 2017) was beneficial.

The workshop was designed and organized around three major ideas:

- 1) Sharpening the interdisciplinary view of the participants. By forming interdisciplinary groups, each participant, while starting from his/her respective comfort zone, had to move away due to the interdisciplinary group setting and the nature of the problem to solve.
- 2) Pursuing a project-oriented approach with practical labs designed for fostering interdisciplinarity, and developing real life projects usable in daily teaching practice.
- 3) Increase networking opportunities and sharing expertise. To reach this goal the participants self-divided into groups requiring maximum diversity in disciplines taught,

³² <https://www.stemawards.eu/>

and schools of provenience. The sharing of experiences was facilitated through participant-led reflection moments.

The student lab sessions were organized into two courses with seven meetings each: one for high school students of technical Computing studies and the other for students majoring in different disciplines. Projects were developed in teams with care taken to ensure interdisciplinarity in each team. In the final competition all the teams were guided through a reflection to quantify and give a concrete perception of the progress done in the learning path. A special mention was appointed to each group in order to award all teams with a concrete gratification of their efforts.

1.7.3 Competitions in Nebraska, Kansas and the USA

1.7.4.1. Nebraska

Nebraska has a rich history of competitive programming that includes contests for high school students and college students. Under the leadership of the University of Nebraska at Lincoln (UNL), many high schools sent teams to the annual Computer Science and Engineering (CSE) Day where they competed in programming contests and in written tests of logic and CT concepts. Top teams were awarded scholarships to be Computer Science/Engineering majors. Colleges and universities from Nebraska and neighboring states sent teams to the annual regional contest of the International Collegiate Programming Contests (ICPC). For many years UNL teams advanced to World Finals competitions. The high school contests served as feeders to the collegiate contests, and the winning tradition motivated all teams to study and practice. For better or worse, CSE Day events ended a couple years ago as the need to recruit more students eased, and UNL has just ceased to host the regional site as priorities for the department have changed, with more attention being paid to other activities such as hackathons.

1.7.4.2. Kansas

1. Kansas, with a particular emphasis on Kansas State University, has a tradition in student participation in programming contests such as the **ACM International Collegiate Programming Contest**³³ and the CCSC programming contest, such as the Central Plains one³⁴. Students' preparation has been organized by a volunteer group of discussion supported by educator experience and learning resources (Arefin et al., 2006), (Skiena & Revilla, 2003).

³³ <https://icpc.baylor.edu/>

³⁴ <https://www.ccsc.org/centralplains/programming-contest/>

1.7.4. Lessons learned and best practices

Competitions are valuable for students, encouraging them to challenge themselves, allowing them to compare with peers from other institutions, make new friendships and moreover learn a lot. Particular care must be made to stress the playful aspect of the competitions, especially in the early stages of education, allowing all students to participate, even the ones that feel themselves not adequate to the competitions. Fortunately, our experience is that in both training and in contests, students naturally recognize the friendly nature of the events. However, it should be noted that some students do not enjoy competitions, however cooperative and friendly they may be. Students from underrepresented populations may also need encouragement to engage in competition.

The nature of traditional programming contests emphasizes precision, speed, algorithmic problem solving, performance under pressure, correctness and testing, communication (in the team), teamwork, coding for efficiency, and endurance, all of which are valuable characteristics of programmers. On the other hand, the problems are tightly proscribed and small scale with minimal user interface development. Alternate formats for competition should be explored and employed in order to provide a more comprehensive learning experience.

Giving to students an overall methodology to afford the competitions, such as starting to look at the products or results of previous editions, besides behavioral habits and ethical principles, can give guidelines useful in many other situations in the educational path and in their life.

1.8. Checking

The literature is extensive on how to assess CT in all of its aspects. In (S.-C. Kong et al., 2019) five perspectives on how to assess CT are summarized. The experiences focus on problem conceptualizations and solution operationalization, cognitive approaches, and interdisciplinary curricula in computational biology. In (S. C. Kong, 2019), the author, according to (Brennan & Resnick, 2012a) where an evolving approach to assessing three dimensions, namely computational concepts, practices and perspectives is presented, reviews and identifies methods to evaluate 9 CT concepts, 7 CT practices and 3 perspectives proposed in research studies. In (Román-González et al., 2019) CT assessment tools are classified as summative, formative-interactive, data mining based, transfer tools, perceptions attitude scales and vocabulary assessment. Among these tools we recall (Moreno-León et al., 2017) for its automated CT assessment, and (Maiorana et al., 2015a) for a formative and summative assessment platform combining the possibility of applying data-mining techniques to log automatically collected data. A test to identify CT is described in (Román-González et al., 2018). A software engineering approach for assessing CT is presented in (Corral &

Fronza, n.d.). The approach is based on definition of a goal, its decomposition into questions and analyzing the answers to these questions and discretizing them into metrics able to collect all the information necessary to construct a solution. Using checking by means of engaging activities both for initial attitude foresight (Cetin & Ozden, 2015), formative and summative assessment is supported by the authors.

Finally a large crowd-sourced repository was carefully curated by means of a combination of researchers' and educators' blind review, peer review, and social review by means of rewarding points, and objective parameters such as number of view and number of downloads, of assessment activities to be intended in a broad sense, ranging from multiple choice questions to educational game and projects has to be developed and shared in the educational communities for free. Projects in this sense have started with (Giordano, Maiorana, Csizmadia, Marsden, Riedesel, & Mishra, 2015b) where the VIVA (Vilnius collaboratively coded and Validated computer science questions/tasks for Assessment) platform was designed and a prototype was built and tested. The platform supported, besides crowdsourcing, multiple competency frameworks used to tag each question to the appropriate competency and where a taxonomy questions/tasks type has been mapped to computational thinking concepts and competency framework. The platform was the seed of Project Quantum (Oates, Coe, Peyton Jones, et al., 2016) an online platform of free, high quality, formative assessment, automatically marked, supporting teaching by guiding content, measuring progress, and identifying misconceptions. Recently a tool, mapping classroom activities to both CT and constructionist learning theory has been proposed in (Csizmadia et al., 2019a).

1.8.1 Lessons learned

A free online bank of formative assessment activities, mapped to curricula and frameworks, supporting multiple pedagogical approaches with automatic tracking of student progress can be beneficial to students and educators. Integrating multiple curricula and frameworks and supporting multiple programming tools, from block bases, like App Inventor and Scratch, and text based, like Java and Python, will allow an international use in different countries and contexts.

In assessment of CT, a combined approach using automatically collected log data from both block-based (Maiorana et al., 2015a) and text-based developing environments (Brown et al., 2018) analyzed by means of data mining techniques coupled with data collect through self-efficacy, self-esteem surveys and formative and summative tests . Attention should be put into detecting the underlying cognitive process starting from the youngest (Zhong et al., 2016).

1.9. Convenience

It is important that a quality education for all guarantees that all students have access to resources and instructions that enable all to fulfil their potential (Ibe et al., 2018), (Wobbrock et al., 2011). Missing this goal means denting access to invaluable resources for members of educational communities. To enable Special Educational Need or Disability (SEND) students to participate, it is necessary to:

- 1) Adhere to both the universal design in learning principles (S. E. Burgstahler & Cory, 2010a)], (Rogers-Shaw et al., 2018) and Universal Design Instruction principles (S. Burgstahler, 2009a)
- 2) Participate and put effort into one of the 25 ways that the Disabilities, Opportunities, Internetworking, and Technology (DO-IT) is changing the world (Dlab et al., 2020) through its programs³⁵.
- 3) Develop accessible resources. Having accessible resources and putting significant effort in developing such resources have a positive effect on the whole educational community. Many tools and resources are available. The Daisy consortium provides a comprehensive list of tools and resources for creation, conversion and validation of accessible publications³⁶ (Park et al., 2019a). In this direction we could list some initiatives and tools:
 - a) Poet Image description tool (Diagram center, 2015b) and their guidelines for describing images, such as the flow chart guidelines³⁷ (Diagram center, 2015a).
 - b) Captioning software like the Caption and Description Editing Tool (Cadet)³⁸, developed by the National Center for Accessible Media (NCAM) at WGBH educational foundation (Foster & Connolly, 2017), (Linebarger, 2000) helps in completing an automated captioned video.
 - c) Interactive video transcripts. Interactivity allows for searching the video using text phrases, running transcripts with synchronously highlighted text, and the possibility to click on the text and jump to the selected part on the video (Wildemuth et al., 2003).
 - d) Tactile reading^{39,40} and digital talking book or spoken book (Argyropoulos et al., 2019) with a crowd-sourced volunteer effort in Europe⁴¹ and around the world⁴². Crowd-sourcing the effort will allow a rich set of books to be available

³⁵ <https://www.washington.edu/doiit/programs>

³⁶ <https://daisy.org/activities/>

³⁷ <http://diagramcenter.org/specific-guidelines-d.html#44>

³⁸ <http://ncamftp.wgbh.org/cadet/>

³⁹ <https://learningally.org/>

⁴⁰ <https://www.mtm.se/en/tactilereading2017/>

⁴¹ <https://www.libroparlato.org/>, <https://adovgenova.com/>

⁴² <https://learningally.org/>

to a wide community beyond dyslexic and people facing visual difficulties. They can serve as great teaching and learning tools in-line courses with the multimedia requirements of 21st century learners.

- e) Use 3D Printing⁴³ for creating tactile experiences.
- 4) Develop accessible hardware and software tools that will assist both in daily practice activities and in educational activities. Listing all the initiatives here would not be feasible. In the Computing domain we could cite some remarkable examples such as Integrated Development Environment (IDE) specifically designed for SEND learners (Milne & Ladner, 2019a).

1.9.1 Experience report: The Master lab for in-service and pre-service and student teachers for special ability certification

In Italy, a significant nationwide effort has recently been implemented to train in-service, pre-service and teacher students for teaching SEND learners (Shinohara et al., 2018a) at all school levels from primary through high school. The effort culminated in multiyear educational initiatives spread nationwide with master courses for teachers. The courses, usually run in one or two years, provide in many cases a nationally recognized certification for teachers of SEND learners.

A laboratory in this master course was designed with these goals:

- 1) Provide the learners with information seeking tools and techniques applied to the domain of interest, eliciting the main reference points and information sources in the domain of interest. Provide an overview on who and what is taught around the world (Kawas et al., 2019a), (Shinohara et al., 2018a).
- 2) Provide the learners with Computational Thinking abilities using tools and techniques applied in an interdisciplinary setting to construct software and hardware tools as well as learning resources useful for people with special abilities. Test and get feedback from the students involved in their daily class activities, asking feedback with others in the same community.
- 3) Promote communication, collaborations and networking among teachers. Reflection and experience reports peer-lead meetings had a positive impact on the whole educational process.
- 4) Organize the activities around a curriculum delineated by special abilities focusing on technologies, pedagogies and content.
- 5) Favor project based pedagogies in group settings.

⁴³ <http://diagramcenter.org/3d-printing.html>

- 6) Leverage on tools such as concept maps (Cañas et al., 2004), (Liu et al., 2011), (Novak & Cañas, 2006) as a design tool useful for teachers and students.

The main difficulties faced during the programme were due to a severe lack of time for the participants. An overbooked schedule and a tight and tough master program contributed to a heavy cognitive overload in all participants, resulting in lost momentum in the educational activities despite the commitment of teachers and the dedication to their students. Self and group reflection using on-line communication means should be pursued for after-experience reflections. Blended and on-line instruction initiatives should be sought to avoid depleting time, energy and resources to teachers simply for reaching the campus location.

1.9.2 How does computing and CT promote convenience and access for all learners within STEAM?

From the above discussions and from the authors' own experiences, the authors would raise the following points:

- 1) Computing tools, software, hardware and unplugged are all essential parts of the teaching practices.
- 2) CT can and should be used by teachers to co-develop with their colleagues simple and effective Apps and tools for solving real-life problems with their students, thereby improving their educational experiences. The developed products need to be tested with the students.
- 3) Involve in the design and development processes all the class members, infusing a culture of inclusiveness.

1.10. Challenges

There are many challenges in computing education. Admittedly far from being comprehensive, we will provide a summary as distilled from research literature, findings from surveys completed by teachers, and from the authors' experiences of some of these challenges deemed most important in relation to the topics presented in this work:

- 1) Equity and inclusion are to be intended in the broadest sense possible. This has been identified as a challenge by most authors in many domains from K-12 to higher education. This challenge has emerged from researchers (Gal-Ezer & Stephenson, 2014) and referenced in careful analysis of teachers' perception in Europe (Sentance & Csizmadia, 2015), (Sentance & Csizmadia, 2017a), the USA (Yadav et al., 2016) and South American countries (Sentance et al., 2020). Equity and inclusion are meant to include in the educational process more students, including women and underrepresented minorities, learners facing socio-economic difficulties, and students with special abilities. Equity and inclusion mean an even stronger substantial effort for developing customizable and accessible learning resources, offering tools accessible to all students, economically sustaining learners. Despite the fact that a lot

- has been done (Guzdial, 2020b) and a lot we have learned from the computing education community, there is still a great deal to be done by educators and researchers to include all learners regardless of their ability (Upadhyaya et al., 2020)
- 2) Supporting teachers and educators, is the cornerstone of the educational process. This support should not necessarily be enforced by rules; another venue of resource should leverage on the profound desire of educators to positively impact the lives of their students. With adequate support from peers, colleagues and the whole educational community, teachers are and will be able to overcome all the difficulties addressed in literature: from lack of subject knowledge (Sentance & Csizmadia, 2017a), when present (Sentance et al., 2020) due to different studies and life experience, to lack of resources offered by schools and institutions and sense of isolation, a feeling that could affect educators from elementary school to higher education (Almstrum et al., 2005). In higher education the major reported challenges span curriculum engagement and retention, and course administration (Deitrick & Stowell, 2019). All these challenges have been addressed and will be addressed involving the educational community and society starting from the local context. Solutions could range from technological support from other institutions. For example, by donation of replaced technologies of data-center, resources that could represent a great solution in smaller situations. Institutions themselves support students with resources, such as by renting hardware, books and material.
 - 3) Understanding the algorithm design, problem solving and programming process (Mcgettrick et al., 2005), transfer this in carefully designed individualized learning resources and assessment activities aimed at developing the competencies needed in a quickly evolving scientific and technological world. This requires a balancing of design and development, latest innovations often tightened to new and quickly evolving technologies with core lasting concepts and rigorous methodological approaches, tests, written and oral examinations, flexibility in exam date with at least a second chance for each semester, and resources and activities aiming at improving students life. Students have to be adequately supported in all educational settings from face to face, to blended, and online context, from developed to developing countries, from urban neighborhoods to rural areas (LeBlanc et al., 2020) scaffolding them in overcoming the many educational challenges they face in computing education (Piwek & Savage, 2020).
 - 4) Engaging and retaining students with success measured both in the ability to attract new students and in strategies to avoid a high attrition rate. All available channels have to be used to meet this goal: from using multiple strategies to prepare and delivering content, to choosing a mix of pedagogical approaches to meet the needs of more learners, to preferring, especially in the first computing course, the simplest and most effective educational technology and tool rather than the most professional

one (Bruce, 2018), and in general by providing “simpler models of computing as a discipline without neglecting the need for mathematical formalism (Mcgettrick et al., 2005) and rigorous scientific approach.

- 5) Continue and sustain the effort of the computing community in recognizing Computational Science “on par with other scientific disciplines” (Gal-Ezer & Stephenson, 2014) granting to all student at all educational levels, from elementary school through higher education, at least one computing course, possibly on an interdisciplinary setting (Amoussou et al., 2010). This still requires harmonizing both pre-university and university instruction where some countries are still lagging in this process.

1.11. Collaboration and Communication

Communication and collaboration competencies are considered important in modern competencies frameworks for students and educators. Work on instructional strategies (C. S. Miller & Settle, 2011) have demonstrated that unstructured study of examples produces better learning than with other instructional practices. Unstructured study is facilitated by collaboration. Many studies (Griffin & Care, 2014), (Care et al., 2012) advocate for using collaborations platforms in the education of all disciplines, besides programming (Wu et al., 2019). Studies have proven the effectiveness of collaborative educational games (Qian & Clark, 2016), (Sung & Hwang, 2013), (Berland & Lee, 2011), (J. Shih et al., 2010) especially when each learner must complete the entire task before starting the collaboration process (Dlab et al., 2020). We advocate for both the usage of collaborative educational games geared through a collaborative problem solving approach, and for project-based learning activities focused on developing such collaborative games. Languages such as APP Inventor offers collaboration components such as a Bluetooth client and server, and cloud-based database functionality that offers an accessible entry point for developing such types of games, leaving ample space for developing an “app with a global impact”⁴⁴. Such applications, activities and projects should develop both collaboration and communication competencies, competencies that should be formally assessed in school, a common practice in European countries, but less common in the USA where formal oral assessment is relatively lacking.

Finally, e-learning (S. C. Kong et al., 2014) and Community of Practices (CoP) such as eTwinning (Vuorikari et al., 2011) supported by information and communications tools are able to support one-to-one learning (Chan et al., 2006) to virtual communities and foster project-based activities performed by students spread across entire continents. These activities are able to sharpen communication and collaboration competencies even with

⁴⁴ <http://appinventor.mit.edu/>

languages different from the mother tongue, empowering students with a vision of different cultures and customs. Such initiatives have the potential to overcome contingent international tension from school to higher education (Chien, 2019).

1.12. Customs

Educational systems and practices are culturally dependent. Even what we recognize as knowledge and skills (i.e. education) are colored by our life experiences. In some systems the emphasis is on rote learning with recitations used for assessment, while in others it is in guiding the students in constructing their own models of how the world works. In some cultures, learning is very individual while in others it is collaborative. Similarly, there is a range of practices from being competitive to being cooperative. These differences may lead to misunderstandings of standards of academic integrity. How cultural and ethical differences impact the teaching and learning of CT, and vice versa, is the subject of this section.

CT is grounded in constructivist educational theory in which the learner does not simply accumulate facts, but takes an active role in building models of subject knowledge. The ability to discern patterns, make inferences, generalize, sequentialize steps, etc. is integral with CT. There are cultures in which the goals of education tend toward making the grade or ultimately getting the job. In these cases, there may be increased impatience, temptation to seek shortcuts, to cut corners, to seek easier more convenient paths to their goals. Attempting to memorize everything is another resort which might work to some extent in some subjects, but is not possible in others including mathematics and Computer Science. The same can be said about learning by trial and error. However, CT is more immediate, being about the process rather than a more distant goal.

The perceived role and status of the teacher vis a vis the student, expectations for the students, classroom environment, assessment strategies, social stratification of the students, family involvement, pressure for personal excellence or cooperation, perception of education as being for technical/job skills vs. liberal arts or research, even the concept of work or job as the means to afford entertainment and sustenance rather than itself being the fulfillment of life's ambitions, all these culturally dependent characteristics have an impact on the receptiveness of students and teachers alike to CT (R. Morelli et al., 2009).

Cheating and other forms of academic integrity violations are defined by the school and understood in the context of the culture and ethics of the institution, the faculty, and the students (Gotterbarn et al., 2018), (ACM Code 2018 task force, 2018), (South Australia, 2020), (Head, 2018), (S. J. Simon & M, 2016). Our purpose is not to judge the ethical rightness or wrongness of academic practices (or mispractice), but instead to consider the impact on learning and inculcation of CT in the students (Bruce, 2018). In so much as such practices and tolerance of them results in students attempting to bypass the higher

functioning of their minds, they are not appropriate for teaching CT. At the same time, it is essential for instructors to carefully consider the constraints on student behaviours, as to whether they are justified for ethical and academic purposes, or if they are imposed primarily because of traditional practice.

Teamwork can take many different forms. In its simplest implementation, all team members are expected to learn and be responsible for all aspects of the team project. This results in comparable learning by all students in the team, and can be accomplished by the team members educating each other from their own individual discoveries and accomplishments. This may or may not be the objective of the learning experience, either case being justifiable depending on the circumstances. In some cultures, and educational environments, it is the practice for teams to have a more distributed learning experience in which multidisciplinary students contribute according to their own expertise, and none are expected to be responsible for the totality of the learning. Again, this may be justifiable depending on the circumstances.

Sometimes advantage can be made of customs of the region. For example, if there is a strong interest in competition (Burguillo, 2010) through the school sports program, that competitive spirit can be leveraged for programming competitions. Or if the students come from a culture of cooperation (D. W. Johnson et al., 2000), (Qin et al., 1995) and sharing, the projects can be created that both build on that culture and simultaneously incorporate features that ensure breadth of learning, and/or cater to the inclination to build for the common social good. Some institutions are located where there is emphasis and/or resources for special needs/abilities learners. These can be seen as opportunities for developing products that serve those populations.

The authors' experience suggests that attention be paid to all the above cultural and ethical considerations both for areas of concern and opportunities for learning (Guillén et al., 2007), (Bruce, 2018).

1.13. Communities of Practices (CoP)

Quality teaching for all requires the best educators that, according to modern competencies frameworks (Bocconi, Chiocciariello, Dettori, Ferrari, & Engelhardt, 2016), (Caena & Redecker, 2019), (Griffin & Care, 2014), (A.C.A.R.A., 2016) have to be equipped with a variegated, complex set of competencies and should be able to cope with demanding in-the-field activities and research for an informed-by-research approach (C.S.T.A., 2016). To cope with all of this, educators need to be sustained and supported by a Community of Practice (C.-K. Looi et al., 2008), (J.-L. Shih et al., 2010) where peers can exchange resources, best practices and collaborate on project proposals and research activities.

Among the CoP and networks of educators focusing on CT and computing, it is possible to recall a few remarkable examples from the variegated international landscape:

- 1) The CAS community (Crick & Sentance, 2011) focuses on “improving the wider perception of computing and its position within the STEM subject area.
- 2) The CSTA community⁴⁵ has the mission to “empower, engage and advocate for K-12 CS teachers worldwide.”
- 3) The eTwinning community (Papadakis, 2016) has its online platform offering educators the opportunity to communicate, collaborate, develop projects, and share resources.
- 4) The Scientix community (Billon et al., 2019) with “its online portal, social media, training and networking events to the community part with an amazing Scientix Ambassadors”.
- 5) The mobile Computer Science Principles Community (Rosato et al., 2017a) is designed to increase the number of schools offering CS courses and to broaden the participation of traditionally underrepresented students such as females and minorities. This community has grown around a Computer Science Pilot course. Other communities grew around other Computer Science Pilots courses (Snyder et al., 2012).
- 6) Eu Code Week developed an inclusive set of initiatives with a European network of ambassadors, and the Coding in your Classroom Now attracted more than 35,000 Italian primary teachers aiming at introducing computing in Italian schools.

13.1 The interplay between small and large CoP: the case of Google4CS@CT

As a model for development and expanding the network of collaborations, the CS4HS@Ct initiative funded by a Google CS for High School grant (today's Educator PD Grants) organized a series of peer-led teachers workshops aiming at developing CT competencies and introducing computing into an interdisciplinary setting. The teachers' workshops were followed by a series of students' workshops where the students were coached in developing computing group based projects showcased in a local competition where each one was selected as a winner with a special mention for the highest quality in his projects. The workshop arose from a collaboration with the mobile computer science principles, and serves as a model for international educational and research collaboration sustained by virtual communications.

13.2 Lessons learned

In developing a community of practice with the goal of engaging all learners to computing in an interdisciplinary setting it is necessary to leverage the following teaching qualities:

⁴⁵ <https://www.csteachers.org>

- 1) Exploit their domain competences. Start from these and develop computing applications on these domains under the guidance of a computing teacher. From this point extend the teachers' comfort zones to include the computing realm, allowing them to advance at their own pace.
- 2) Build on the existing ability to assess student competence. Let the teachers participate in grading computing artifacts, guided by a solution developed by computing educators.
- 3) Augment the teachers' self-esteem in their ability to act as trainers.

1.14. Citizenship

Today's digital technologies offer opportunities in all aspects of our daily life, and it is important to teach students how to be responsible digital citizens starting in primary school (Ribble & Bailey, 2007), educating them to be part of a sustainable technology development (Giordano & Maiorana, 2013a).

Humanitarian Free and Open Source Software (HFOSS) (H. J. Ellis et al., 2015a), (R. Morelli et al., 2009) collects projects designed and developed with a social benefit in mind. Projects are developed in the fields of health care, disaster management, accessibility assistance, economic development, education, and other areas of social need. The approach can be used in the daily teaching practice of students, including non-computing majors, in all types of activities, from commenting and documenting their projects, reading the code, reverse engineering it or contributing to existing projects or developing new ones. Activities and real life humanitarian projects that can be used from the first day of classes and activities in this regard can be found in (Goldweber et al., 2019). HFOSS projects represent a way to tackle the real challenges, i.e. addressing the real life challenges, such as granting jobs to the students, allowing them to spend their time learning while working instead of looking for jobs that are disconnected from their learning path. Too many times the lack of resources for students results in a real depletion of diversity to the computing and educational community.

1.15. Conclusions

In this work the authors, with the support of research literature and their educational experience, have presented, compared, and contrasted the similarities and differences in educational perspectives in two continents, Europe and America, in regard to CT and the interplay with respect to Content, Pedagogies and Technologies and how CoP can sustain the whole computing community which is composed of learners and educators. After reviewing the main tools that can be used, we shared the main lessons learned from the study of the literature and from our educational experience regarding the capability of CT to foster and sustain the development of 21st computing competencies, creativity, collaboration

and communication, and digital citizenship and how CT can be fostered through competition, and assessed. The main conclusions are the following:

- 1) The need for educators to develop accessible and inclusive resources, content, pedagogical practice and technologies, and differentiated learning trajectories suitable to all learners and differentiable to their educational needs. To reach this ambitious goal the richest set of tools and activities, including formative assessments, educational games, and competitions, must be used.
- 2) Despite the effort already in place, there is still space to expand the exposition to computing in an interdisciplinary setting to all learners from Kindergarten through higher education. A rationalization of the curriculum for non-majors in many countries and in international settings should couple the effort to introduce computing in schools at all levels, especially where some countries are still lagging.
- 3) All students should be exposed to ethics during all learning path.

Chapter 2: International comparison of intended and enacted curricula

Objective of the chapter: Compare and contrast intended and enacted computing curricula across the world through a designed and validated teacher survey instrument

Approach: a mixed method design process centered around the design of templates to compare intended and enacted curricula, design, conduct, analyze and validate a teacher survey

Result achieved and novelty: we were able to provide early observations around aspects on intended and enacted curriculum descriptives. The validated teacher survey instrument, i.e., MEasuring TeacheR Enacted Computing Curriculum (METRECC) instrument.

Significance for the state of the art and the narrative of the work: it provides a solid research foundation framing the successive steps. The METRECC instrument not only captures country level reports of intended curriculum, but also enacted curriculum directly from teachers.

The content of this chapter is a summary of the main finding of country reports on the analysis of data collected through country reports and a teacher surveys related to intended, i.e., policy tools as curriculum standards, frameworks, or guidelines that outline the curriculum teaches are expected to deliver (Falkner, 2019b), (Porter, 2001), and enacted curriculum, i.e. actual curricular content taught by teachers that students engage with in the classroom (Falkner, 2019) laying a solid foundation to guide the reflection of the design, development and assessments of the enacted curricula presented in chapter 4. The details of the country report template and the teacher survey instrument design development and validation were undertaken by Working Group 6 "An International Benchmark Study of K-12 Computer Science Education in Schools" and the report crafted for the ITICSE 2019 Conference (Falkner, Sentance, Vivian, et al., 2019d). The work is the result of an international collaboration spanning three continents across the globe as shown in figure 1.



Figure 1: International rich of the ITiCSE 2019 working group (courtesy of Christine Liebe & Monica M. McGill)

The Working Group reviewed and analyzed pilot data from 244 teachers across seven countries (Australia, England, Ireland, Italy, Malta, Scotland and the United States). We analyzed the pilot results (n=244) and applied four validity tests: face validity, concurrent validity, population validity, sampling validity and construct validity, in addition to a focus group to further revised the instrument. The report presented the pilot results and outcomes of validity testing, as well as revisions made to the instrument. The resulting METRECC tool combines a country report template and a teacher survey that will provide K-12 teachers with a means to communicate their experience enacting CS curriculum. National and regional policymakers can use METRECC data to inform iterative curriculum revision and implementation. We provided open access to the METRECC instrument and data set.

2.1 Objectives and research questions

The broad objectives of the Working Group were to:

- 1) To build an international research collaboration and strategy for measuring K-12 CSED implementation in schools.
- 2) To initiate a scalable, collective effort for a deeper investigation into what is happening in schools, based on the experiences of educators in classrooms.
- 3) To develop an open source teacher survey instrument that can be implemented across countries

The following research questions guided the work of the working group:

- 1) What are the similarities and differences across countries in terms of intended CS *curriculum topics and programming requirements?*

- 2) To what extent are teachers *addressing the intended CS curriculum with their enacted curriculum* in classrooms?
- 3) Create a template for capturing the required curricula, standards, and policies in place for country or state
- 4) As a pilot study measure the enacted curricula through the development of survey instrument from the teachers' perspective

2.2 Methods

This study adopted a mixed-methods design process centered around the development and evaluation of a teacher survey instrument, that included a review of related K-12 CS survey instruments and development of instruments for this study, along with a pilot of the instruments and a focus group to revise the teacher survey instrument. This process is supported by approaches in educational and psychological testing (Hubley & Zumbo, 2013) that use a combination of theory and expert opinion as the basis for the development and selection of testing items, paired with an iterative and multi-stage process in evaluating test items (in this case being teacher survey items). In the following sections we describe the processes involved in developing the two instruments used in this study: the country report and the teacher survey.

2.3 The country reports

To develop the country report template, a number of reports and papers capturing international and regional data were used as a basis to identify potential key categories relevant to comparing and contrasting school demographics and intended CS curriculum across countries, e.g. (Hubwieser et al., 2015), (Hubwieser et al., 2011), (R. Society, 2017), (Sentance & Thota, 2013a), (Code.org, 2018). TheWorking Group searched and curated relevant papers from the ACM Digital Library and Google Scholar. Search terms such as "informatics", "computing", "digital technologies", were included to capture reports for countries referring to CS curriculum in alternative ways. The reference list of the identified papers were used as a basis to identify other key papers. These references were curated into a spreadsheet with details entered for each of the headings (e.g. date of publication, year levels, country, methods, etc.). Papers were included if they captured or reported on country or multinational K-12 CS education from an intended curriculum perspective (e.g. details around topics, age bands). From here a new spreadsheet was devised to curate key categories and survey item questions from these prior studies to form a country report template.

There was a challenge in capturing implementation of CS topics across countries, due to the differences of CS curriculum between countries as well because it was dependent on whether a specific CS curriculum was available. Therefore, it was decided that a comprehensive measure of CS topics being implemented was a key consideration of the enacted curriculum

and would be captured via the survey instrument. However, as a broad comparison across countries, we reviewed various curriculum analysis reports (Mannila et al., 2014a), (Barendsen et al., 2016), previously mentioned country reports, country curriculum documents (of those represented by Working Group Members) and the CSTA standards (Seehorn et al., 2011). We used curriculum documents and CSTA standards as a starting point of broad CS topics which were expanded on by the Working Group for a high-level comparison across countries.

The recurrence of CT within literature merited its inclusion as a high-level topic. The goal is that the METRECC instrument would seek to identify more specific and fine-grained topics which would be used to inform the revised country report instrument. All the curated categories were organized in a spreadsheet and presented as a template to be completed by survey administrators. The Working Group reviewed the draft country report template to determine which categories would be eliminated, adapted or kept, taking into account considerations toward language, nuances and transferability across countries. Items which the group deemed difficult to clarify were removed. For example, the provision of national funding (taken from (Codeorg, 2018)) was eliminated due to identification of the vast differences across countries funding schemes.

Additionally, in recognizing the challenge of mapping CS curriculum availability and implementation requirements across countries due to differing age groups for grades, it was decided that student ages would be included alongside grades for ease of completion. To test the template, each Working Group member took the template and completed it for their respective country. As members completed the template, they noted any confusion around language, categories or problematic categories. No significant changes were made but it was decided that to support administrators to complete.

the country report template, instructions and a glossary would be provided.

The following information is captured in the country report template:

- 1) Country demographics and information relating to schools (e.g., total population, number of schools, number of teachers).
- 2) CS curriculum state or country plan standards and requirements.
- 3) Year Level (with age for comparisons) mapped to prescribed curriculum and programming requirements.

Table 1 reports an overview of the educational system in the working group authors' countries.

Table 1 Demographics of Authors'(working group 6) countries/states education systems highlighting links to Computing Science Standards (CSTA) July 2019

COUNTRY/ USA STATE	AUSTRALIA (AUS)	COLORADO (US-CO)	ENGLAND (ENG)	IRELAND (IRL)	ITALY (ITA)	ILLINOIS (US-IL)	MALTA (MLT)	MINNESOTA (US-MN)	SCOTLAND (SCO)
Population (million)	25.09	5.69	55.62	4.70	60.50	12.7	0.47	5.6	5.44
No. of schools	9477	1900	29972	3961	8636	4266	170	2066	2400
No. Primary schools				3246			108		2031
No. secondary schools				715			62		359
No. of students	3893834	911536	8378809	920867	8422419	2072880	46247	862971	693251
No. of teachers (FTE)	288583	59989	498100	66327	872268	135701	2976	57262	51959
No. of Primary teachers (FTE)				36773					
No. of secondary teachers (FTE)				29554					

Table 2: Contry computing overview

COUNTRY/US A STATE	AUSTRALIA (AUS)	COLORADO (US-CO)	ENGLAND (ENG)	IRELAND (IRL)	ITALY (ITA)	ILLINOIS (US-IL)	MALTA (MLT)	MINNESOTA (US-MN)	SCOTLAND (SCO)
CS State or country plan	√	⊗	√	⊗	∅	⊗	√	⊗	√
CS Curriculum k-6 standards defined	√	⊗	√	∅	∅	⊗	√	⊗	√
CS Curriculum: Y7+ standards defined	√	√	√	∅	∅	√	√	⊗	√
CS Guidelines - standalone subject	√	√	√	∅	∅	⊗	∅	∅	√
CS Guidelines - across disciplines	⊗	⊗	⊗	⊗		⊗	∅		⊗
Teacher autonomy to implement state/country guidelines as standalone or cross discipline	√	√	√	⊗		√	∅		√
CS Formal Reporting	V	⊗	⊗*	⊗	∅	⊗	∅	∅	∅
CS in pre-service training Primary	E	E	√	E	√	E	⊗	⊗	E
CS in pre-service training Secondary	E	E	√	E	E	E	√	⊗	√
CS training for inservice Primary?	V	√		V		⊗	√	√	
CS training for inservice secondary?	V	√		V		√	√	√	
Year endorsed	2015	2018	2013/14	⊗	⊗	⊗	2018*	⊗	2016*

Table 3: Intended curriculum

CONCEPTS COVERED	Intended Curriculum (Of those countries/states with state plan)								
COUNTRY/US A STATE	AUSTRALIA (AUS)	COLORADO (US-CO)	ENGLAND (ENG)	IRELAND (IRL)	ITALY (ITA)	ILLINOIS (US-IL)	MALTA (MLT)	MINNESOTA (US-MN)	SCOTLAND (SCO)
CONCEPTS COVERED	Intended Curriculum (Of those countries/states with state plan)								
Computational Thinking	√P √S	⊗P√S	√	⊗P √S	⊗P √S	⊗	√P √S	⊗	√P √S
Computer Systems	⊗P √S	⊗P∅	∅	⊗P √S	⊗P √S	⊗	⊗P √S	⊗	⊗P √S
Networks and Internet	⊗P √S	⊗P∅S	√	⊗P √S	⊗P √S	⊗	⊗P √S	⊗	⊗P √S
Data & Analysis	⊗P √S	⊗P√S	√	⊗P √S	⊗P √S	⊗	⊗P √S	⊗	⊗P √S
Algorithms and Programming	√P √S	⊗P∅S	NA	⊗P √S	⊗P √S	⊗	⊗P √S	⊗	√P √S
Impact of Computing	√P √S	⊗P√S	√	⊗P √S	⊗P √S	⊗	⊗P √S	⊗	√P √S
Other areas not covered above									

(i) Yes (√) No (⊗) Additional information (∅) (ii) Pre-service training - Varies(V)

Compulsory (√), Elective (E)

(iii) CSTA standards covered Explicit (√) Implicit (∅) Not covered (⊗) *Date previous CS

Australia: CS curriculum implementation is at the early stages of implementation with each state or territory determining reporting requirements. As a result, reporting expectations vary between schools, this applies to both government and privately funded schools. Formal pre-service training and inservice professional CS learning varies in terms of requirements and availability. No national curriculum is mandated at the final stages of secondary school (Grade 11 and Grade 12) because courses are optional for students and align to final certification.

Colorado: In Colorado each district decide whether or not the curriculum is required. defines the CS standards and each high school decides if CS is a standalone subject or delivered across subject areas. If there is no district wide curriculum defined then primary and middle schools have autonomy to implement the CSTA standards. In addition, some charter schools offer STEM integrated education At the time of writing approximately 30% of schools offer CS, therefore, many students do not have access to CS education. State funding is available for CS inservice professional development.

Illinois: In Illinois, currently there are no state standards but districts have the ability to implement their own. Chicago public schools, for example, implemented a graduation requirement that all high school students have one year of computer science education.

Ireland: In Ireland students attend primary school until the age of 12. Their secondary school education is in two phases firstly the Junior Cycle at age 12-15 (which included first, second and third year, where first year is entry into second level) followed by the Leaving Certificate (which includes fifth and sixth year). These phases/years are mandatory across all schools. There is an optional year, TY (also known as transition year or fourth year) which is an optional year (but some schools make in mandatory as part of a local arrangement). In the Junior cycle students undertake short courses across a range of subject areas which includes an optional coding. In 2018 Ireland finalised the pilot upper secondary CS curriculum. By September 2020 all schools will be eligible to implement the CS curriculum. However, this is optional for schools to decide if they will deliver this throughout the session. In primary, the CS curriculum is under development. The pilot phase involved a 'bottom up' approach with school implementing 'rough topics' based on the findings they will develop the curriculum. This is expected to be rolled out

by around 2022. Although the secondary curriculum s optional teachers have control within that to decide on resources and pedagogy.

Italy: In Italy the secondary school system has a range of high schools eg: art, classical, scientific, technical, linguistic and vocational. Computing science is delivered in technical and some science high schools. There is CS guidance for primary and middle school where computational thinking is suggested to be taught. In High school there are national guidelines for the majority of liceo and have introduced for vocational study too. Formal reporting takes place some high school . Kindergarten is mandatory from age 3. There is experimentation to do four year at high school to align with the European Community. CS is not mandatory in all types of higher high school and it is suggested in primary and lower high school. OOP is mandatory in some technical higher high school.

Malta: From 2018-2019 all pupils from Year 7 to Year 11 follow a ICT C3 certificate which includes CS education. In primary the CS learning objects are cross curricular and the teache decides how and when implemented. These are not assessed. CS is a standalone subject at Y9. There are two branches one is networking and vocational/hands on. The other branch includes programming, databases, computer architecture. Formal CS reporting is in secondary schools Y7 to Y11. Pre-service CS training is compulsory for teachers delivering CS from Y7-Y11. In 2018 the original ICT C3 curriculum was updated.

Minnesota: Each school district decides if CS is a standalone subject. The state government is trying to include computational thinking within the performing arts and science standards revisions. Although students do not experience a state mandated CS curriculum, teachers can choose to incorporate CS into their classrooms.

Scotland: All pupils have an entitlement from pre-school up to 3rd year in secondary school to a Broad General Education which included computing science as a standalone subject which can be teachers and schools have ownership on its delivery. 4th year to 6th year computing science is optional for qualifications. In 2016 the computing science curriculum k-10 "Broad General Education for curriculum content for computing science was refreshed"

Although CS training is available for primary and secondary teachers all WG 6 members reported that this training is optional and variable.

Table 4: Approximate age and school placements across authors' countries/states education systems.

COUNTR Y/	AUS	ENG	IRL	ITA	MLT	SCO	USA
USA STATE							
AGE* (Years)							
2+				Pre-school	Kindergarten	Pre-school	Pre-school
3	Pre-school	Pre-school	Pre-school	Kindergarten	Kindergarten	Pre-school	Pre-school
4	Kindergarten	Pre-school	Junior Infants	Kindergarten	Kindergarten	Pre-school	Kindergarten
4 – 5	Reception/foundation	Reception	Senior Infants	Kindergarten	Year 1	Primary 1	Grade 1
5 – 6	Year 1	Year 1	First Class	First class primary	Year 2	Primary 2	Grade 2
6 – 7	Year 2	Year 2	Second Class	Second class primary	Year 3	Primary 3	Grade 3
7 – 8	Year 3	Year 3	Third Class	Third class primary	Year 4	Primary 4	Grade 4
8 – 9	Year 4	Year 4	Fourth Class	Fourth class primary	Year 5	Primary 5	Grade 5
9 – 10	Year 5	Year 5	Fifth Class	Fifth class primary	Year 6	Primary 6	Grade 6
10 – 11	Year 6	Year 6	Sixth Class	First class lower high school	Year 7	Primary 6	Grade 7
11 – 12	Year 7	Year 7	First Year	Second class lower high school	Year 8	Primary 7	Grade 8
12 – 13	Year 8	Year 8	Second Year	Third class lower high school	Year 9	S1	Grade 9
13 - 14	Year 9	Year 9	Third Year	First class higher school	Year 10	S2	Grade 10
14 - 15	Year 10	Year 10	TY	Second class higher school	Year 11	S3	Grade 11
15 – 16	Year 11	Year 11	Fifth Year	Third class higher school	Sixth form lower	S4	Grade 12
16 - 17	Year 12	Year 12	Sixth Year	Fourth class higher school	Sixth form higher	S5	
17-18				Fifth class higher school		S6	

*Youngest age at the start of the school session. For example, in Scotland for session 2019/2020 almost all children between the ages of 4 and a half and 5 years old will start primary school at the start of term in August. Children who attain the age of 5 years between 1 March 2019 and 28 February 2020 should be registered for education in January 2019 to start school in August 2019.

2.4 The teacher survey

The Working Group undertook a collaborative, iterative process to develop a teacher survey instrument that could be transferable across countries. The Working Group broadly undertook the following steps to define the survey categories and questions:

- 1) Curation and review of CS and education survey papers and reports, identifying those that included survey instruments with evidence of reliability and validity.
- 2) Identification of survey categories.
- 3) Curation of survey questions from surveys with reliability and validity evidence that aligned with survey categories.
- 4) Addition of new survey questions for categories that were not found in surveys with evidence of reliability and validity.
- 5) Refinement of survey categories and questions and selection of questions for inclusion in the survey.
- 6) Building of the online survey and final survey reviewed by all members.

The Working Group leaders developed a set of key categories that might be of interest internationally as a starting point. The categories were shared with Working Group members for review, alterations and the addition of new categories. Although initially seeking to identify CS education surveys and articles reporting on teacher surveys, the search was broadened to also review known international education survey instruments such as the TALIS Survey (Ainley & Carstens, 2018) that could provide valuable survey items with evidence of validity and reliability for demographics and teaching practices. Once a set of draft categories were agreed upon by the Working Group, these formed separate sheet labels in a Google sheet. Collaboratively, Working Group members curated and added questions from surveys with evidence of reliability and validity, including identifying metadata such as the "sub-category" (e.g. classroom equipment), "response options" (e.g. laptop, computer, tablet, other), the "source", e.g. TALIS Survey (Ainley & Carstens, 2018), the measure (e.g. Likert, checkbox, multiple choice), possible threats to validity and whether or not the questions were from a survey instrument with previous evidence of reliability and validity. This resulted in 88 initial example questions from 11 sources (Ainley & Carstens, 2018), (Bandura, 2006), (D'Anca, 2017), (Dweck, 2006), (Jormanainen, 2018), (A. C. Porter & Smithson, 2001b), (Quille & Bergin, 2019), (Stupnisky et al., 2018).

Upon developing the questions, consideration was taken with regards to the best way to measure responses. Here, we discuss some examples and how prior survey instruments that

have evidence of validity have been utilised. To capture teacher demographic data and teachers' classroom composition in Sections two to four of Table 5 (e.g. gender, low-socio-economic status, disability, gifted students), we adopted a majority of TALIS (Ainley & Carstens, 2018) questions as these have been found to translate across 48 countries. For classroom composition, we use teachers' estimations of how many students have various characteristics against a percentage. We also utilized TALIS questions and items about professional development activities and barriers for section 10 relating to professional development.

This allows us to compare benchmark results against TALIS survey reports and also allow administrators to compare estimations against their country report breakdowns. To better understand teachers' instructional practices, we reviewed questions in works by (Ainley & Carstens, 2018) and (A. C. Porter & Smithson, 2001b), as the authors provide guidance around capturing classroom practice. (Ainley & Carstens, 2018) recommend using frequency of instructional practices rather than measuring teachers' agreement towards the adoption of practices. Similarly, (A. C. Porter & Smithson, 2001b) invite teachers to estimate and nominate time spent against various instructional practices in terms of percentage of implementation (e.g. 25-49% on "whole class instruction"). The authors' reason that this measurement facilitates comparisons across classrooms, types of courses, and types of student populations and that they have the advantage of being easy to respond to (i.e., in cases when teachers teach multiple classes or for helping teachers reflect on time spent against practices as they can estimate using various time measures, such as a week or a year of instruction). However, a major disadvantage is that such measures provide a crude estimate. To reduce complexity, we did not include the full matrix columns by (A. C. Porter & Smithson, 2001b) (A. C. Porter & Smithson, 2001a) that invited teachers to reflect on practices across Bloom's Taxonomy items.

Some items from the Research-Practice Partnerships CS For ALL (RPPforCS) Survey Instruments (CSForALL, 2019) were adopted in section 3 around teachers' current work and section 10 inquiring about their professional development and use of professional development materials in the classroom. The RPPforCS project collects participation data about teachers participating in the CS for All: Research Practitioner Partnership Project. RPPforCS is focusing on the projects preparing teachers to offer a stand-alone high school course in CS, however, they have made their instruments available to support others in capturing CS implementation.

The survey component measuring CS self-esteem utilised the Bergin Self-Esteem Instrument (Bergin, 2006) that was developed as part of a longitudinal study, also utilised by (Quille & Bergin, 2019) with CS student cohorts. Bergin had developed the instrument as a modification of the Rosenberg self-esteem scale, which has generally been shown to have evidence of high inter-item and test-retest reliability evidence (Rosenberg, 2015) to apply to programming. The 10 items used in the Bergin (Bergin, 2006) study were added to the

instrument, however, the domain-specific subject was adapted from "programming" to "Computer Science" to reflect the broader K-12 CS curriculum that the survey was investigating.

Teachers responded to statements on a 7-point Likert scale, from "Strongly Disagree" to "Strongly Agree". The items were generally about CS capabilities and we wished to measure teachers' self-esteem to determine how impact on classroom practice. Some additional question development were devised using survey instruments used in other studies, such as by (Vivian & Falkner, 2018), and those developed as a collaborative exercise by Working Group members.

Our Working Group investigation to evaluate this teacher survey instrument will involve checking these questions for evidence of validity. The Working Group held an online meeting in which the group worked through the curated questions to determine whether to "keep" or remove them as well as considering and discussing the language of questions, duplicates, response options and the transferability of questions across the various countries and alignment with the study objectives (e.g. to investigate the enacted curriculum). This process was undertaken twice (once offline) and resulted in the final set of categories (now referred to as sections) and 51 key research questions that were ready for import into the SurveyMonkey tool. A number of sub-sections and questions, particularly within teacher confidence and motivation, were excluded from the final survey. The final draft survey was downloaded from SurveyMonkey as a PDF and emailed to Working Group members for review, with required amendments made. Two researchers tested a copy of the digital survey on SurveyMonkey. The final survey instrument resulted in 11 sections with 11 pages and 53 questions (two questions being administrative). The survey overview is presented in Table 5. 33 (58.5%) of the total survey questions were set as required" with the remaining as optional. Required questions were determined as those key to answering our Working Group research question that focused on the enacted curriculum, with optional being extensions and as useful to providing additional supporting data. In the following section, where relevant, we broadly describe some of the survey sections and where questions and measures were sourced from. A final question asked teachers if they would be willing to consent to their anonymous data being shared with the computer science education research community for future use.

A large portion of survey questions (39.6%, n=21) related to teacher demographics, their current role and qualifications/experience (see Table 5). The second highest portion of questions related to what teachers are doing in the classroom and the resources and practices they are adopting (39.6%, n=21), aligning with our survey goal of investigating the enacted curriculum. Additional question topics related to student cohort composition, professional development and teacher' perceived CS self-esteem. Examining an overview of the types of questions utilised in the survey instrument, there were a reasonable split between multiple choice questions (35.8%, n=19) and matrix questions (34.0%, n= 18) that used Likert

style. Details of the data analysis section can be found in (Falkner, Sentance, Vivian, et al., 2019d).

Table 5. Survey overview

Section	Sub-topics	Question numbers	Total questions (required)	%
1. Introduction	Study information Consent to participate	1	1	1,9%
2. Demographics	Teacher demographics (e.g. age, location) School demographics (e.g. socioeconomic, remoteness)	2-11	10	18,9%
3. Current work	Employment Teaching role Subject expertise Experience teaching CS	12-18	7	13,2%
4. Qualifications	Qualifications in teaching, computing and other subjects Participation in classroom research	19-22	4	7,5%
5. Student composition	Student cohorts Classes taught and class size Demographics of students (reported)	23-25	3	5,7%
6. Support and resourcing	Access to infrastructure, facilities and equipment School support (people, PD) and perceived needs Place of CS classes Local CS outreach engagement and awareness CS topics taught and unplugged/plugged Curriculum document/s used (if any) Access to CS and general teaching materials and technology	26-38	13	24,5%
7. Assessment of student learning	Implemented assessment approaches in CS Reporting required or not	39-40	2	3,8%
8. Classroom practice	Learning and teaching strategies (CS specific and general) Programming environments and motivation for use	41-46	6	11,3%
9. Self-efficacy and confidence	Teachers' perceptions of their CS capabilities	47	1	1,9%
10. Professional development	Participation in types of PD activities Structure/benefits of PD activities Perceived PD needs Extent PD resources used in classroom	48-52	5	9,4%
11. Open access data	Consent for anonymous data to be included in open access	53	1	1,9%
			53	100,0%

2.4.1 Intended Curriculum Observations

Within the pilot sample, England was the first to endorse a CS curriculum in 2014. To date, formal curriculum (or standards/ frameworks) have been endorsed across all countries, except Ireland, Italy and in the US where it is state-dependent. Although some countries have national CS curricula, there are observed variations regarding formal reporting of student learning outcomes in CS. In a study of Australian teachers (Vivian & Falkner, 2018), it was found that teacher self-efficacy increased with formal reporting requirements as teachers had developed more experience in assessing student learning. This suggests that this is something worth investigating and monitoring across countries. Although some locations, such as Colorado, England and Malta, indicated that they have compulsory CS training for primary and secondary teachers, it is clear that this is not something that has been standardised across other regions, irrespective of a formal curriculum being introduced. Additionally, pre-service teacher training is only provided in England, with a majority of other locations having this as an optional study elective at this stage. Findings from the Working Group responses categorise the CS curricula into three broad types: those with a state plan for CS in place, those with no state plan for CS in place and those whose CS state plan is in development. CS guidance for those with a state plan was either through standalone delivery or embedded across disciplines. All teachers have flexibility of implementation within their state plan curricula. They all have the opportunity to plan delivery of lessons and choose resources. Lesson structure, delivery and content is not prescribed. All of the countries, except the US, cover some aspect of the CS concepts presented in Table 6. Within the US, we can see that Illinois does not cover any of the concepts explicitly but Colorado does. In four out of the nine countries/states with a K-6 national/ plan, all cover "Computational Thinking", "Algorithms and Programming" and "Impact of Computing". In the seven countries/states with a state plan for students Year 7 onwards, curriculum concepts include "Computational Thinking", "Computer Systems", "Networks and Internets", "Data and Analysis", "Algorithms and Programming" and "Impact of Computing". We observe that in Table 6 some countries have defined programming languages that are to be taught at specific year levels and others have not. Australia, England, Italy and Scotland have all defined programming languages from primary years of schooling. Programming Languages for those countries/states with a national/state plan use Visual Programming through K-6. From Year 7 onwards, General Purpose Programming is used, moving to OOP in later grades.

Table 6: Programming language curriculum specification across pilot study states and countries.

Age at the start of the school session

Not specified (X) Visual Programming (VP) General Purpose Programming (GPP) Object Oriented Programming (OOP)

Age* / US Grade	AUS	US-CO	ENG	IRL	ITA	US-IL	MLT	SCO
3-4 Pre-school		X				X		VP
4-5 Pre-school	X	X				X		VP
5-6 Kinder.	X	X	VP	X	VP	X	X	VP
6-7 Grade 1	X	X	VP	X	VP	X	X	VP
7-8 Grade 2	VP	X	VP	X	VP	X	X	VP
8-9 Grade 3	VP	X	VP	X	VP	X	X	VP
9-10 Grade 4	VP	X	VP	X	VP	X	X	VP
10-11 Grade 5	VP	X	VP	X	VP/GPP	X	X	VP
11-12 Grade 6	GPP	X	GPP	VP/GPP	VP/GPP	X	VP	VP/GPP
12-13 Grade 7	GPP	X	GPP	VP/GPP	VP/GPP	X	VP	VP/GPP
13-14 Grade 8	OOP	X	GPP	VP/GPP	VP/GPP/OOP	X	GPP	VP/GPP
14-15 Grade 9	OOP	X	GPP	VP/GPP	VP/GPP/OOP	X	GPP*	GPP
15-16 Grade 10	X	X	GPP	VP/GPP	VP/GPP/OOP	X	GPP*	GPP
16-17 Grade 11	X	X	GPP	VP/GPP	VP/GPP/OOP	X		OOP
17-18 Grade 12		X	GPP/OOP	VP/GPP	VP/GPP/OOP		X	

2.4.2 Enacted Curriculum Observations

Due to the small sample size of this pilot study and the newness of K-12 CS education, we aggregate the reporting of teacher demographics and descriptives as a model for exploring this data. However, we acknowledge that teacher preparation, expectations, and experiences in CS varies across across grade bands (i.e., primary, middle years, and secondary) may impact on enacted CS curriculum. Survey administrators can choose to break up the reporting of grade bands. Future research using the survey will investigate results across these grade bands for more useful and detailed reporting. Tables 7, 8, 9, and 10 provide insight from the survey on what teachers' enacted curriculum looks like across countries, including what they are using to teach CS, what resources they perceive is needed to teach CS, who they are teaching, and what they are teaching. For instance, all seven countries were similar in that the most common workplace equipment teachers used to teach CS were desktop (n=183) and laptop (n=128) computers, with smart phones being the least utilised, if at all (see Table 7). Although Table 8 shows the two most commonly selected needed resources across

countries were classroom lesson resources (n=136) and professional development (n=124), looking more closely at individual countries responses illustrates a wide array of contexts. For example, teachers in England and Scotland selected wanting more CS-specific technology (n=30 and n=14) more often than other resources, compared to Malta who selected non-CS specific technology equipment as their most common needed resource. Across the countries, the most common areas of expertise teachers selected were computer

science (n=180) and ICT (n=111), followed by Math and numeracy (n=66) (see Table 11). Some teachers stating that they did not teach CS may actually teach aspects of CS. Possibly, the courses they teach are called STEM, robotics, or something else, but they include aspects of programming and computational thinking into their course. The levels teachers are instructing are also important to consider when looking at teacher pedagogy and what occurs within the classroom, as well as when making comparisons to intended curriculum, since it often varies depending on the level and/or age range of students. Across countries, lower secondary (13-15 years old) (n=156) and secondary (16-17 years old) (n=150) teaching year levels were the two most common selections (see Table 9) in our pilot survey. In contrast, more respondents from Australia work with upper primary (11-12 years old) while Italy had more teachers working with secondary and senior secondary (18-19 years old) levels. The variety in student levels also helps to explain the breadth visible in the content taught (see table 10). Across countries, programming skills and concepts (n=219) and algorithms (n=204) were the most frequent CS content being taught by teachers whereas machine learning (n=46) and artificial intelligence (n=67) were the least commonly taught (see Table 10), reflecting some of the similarities in the intended curriculum across these seven countries. There are some differences in enacted curriculum across countries, for example, robotics, which is taught by a higher percentage of teachers in US and Australia than other countries. Moving forward, we anticipate that breaking down the content taught by each country and comparing it to the intended curriculum there can highlight how teachers are enacting and perceiving the intended curriculum, and provide some focus for future resource development. Further analysis and filtering of the data by year level may provide additional insights into the enacted curriculum, as well as offering a way to make connections to each country's intended curriculum.

Table 7: Frequency of reported classroom equipment usage by country

Country	Laptop	Desktop	Tablet	Smart Phone
US	64	78	38	16
England	17	44	15	2
Italy	10	15	6	6
Ireland	12	17	8	4
Scotland	11	18	5	3
Australia	11	6	10	
Malta	3	5		
Total	128	183	82	31

Table 8: Frequency of teachers reporting needed resources by country.

Country	Non-CS specific technology equipment	CS-specific technology	Improved technology infrastructure	Professional development	Classroom lesson resources	CS Professional Mentor	School collaboration	Support for classroom research
US	26	52	29	52	62	35	43	29
England	13	30	16	27	24	18	20	24
Italy	4	6	6	10	7	6	7	6
Ireland	5	8	6	13	14	10	9	5
Scotland	8	14	12	10	13	3	7	3
Australia	1	4	5	10	13	7	5	6
Malta	4	2	3	2	3	2	1	3
Total	61	116	77	124	136	81	92	76

Table 9: Frequency of teachers teaching at year level bands by country.

Country	Pre-primary (3-5 years old)	Junior Primary (6-7 years old)	Pri-Primary (8-10 years old)	Upper Primary (11-12 years old)	Lower Secondary (13-15 years old)	Secondary (16-17 years old)	Senior Secondary (18-19 years old)
US	5	15	17	36	76	70	56
England	5	10	12	33	38	36	27
Italy	1	2	4	2	8	14	13
Ireland		2	5	7	13	13	8
Scotland	1	1	2	9	16	15	6
Australia	3	6	9	12	3	1	
Malta		2		2	2	1	1
Total	15	38	49	101	156	150	111

Table 10: Frequency of teachers reporting content taught by country

Country	Programming skills and concepts	Algorithms	Cybersecurity	Robotics	Artificial Intelligence	Machine Learning	Networks and Digital Systems	Information Systems	Web Systems	Hardware	Ethics	Data representation	Privacy	Databases	Data analysis and visualisation	Computational Thinking	Design process (or Design Thinking)
US	100	94	66	54	34	24	52	40	44	70	86	78	74	31	50	83	83
England	52	52	43	17	23	12	47	30	32	47	46	46	40	37	23	50	28
Italy	16	14	7	8	2	1	8	6	10	11	7	9	8	9	5	9	4
Ireland	19	13	3	8	6	5	3	4	7	13	11	10	8	8	5	13	11
Scotland	18	18	13	2	1	2	7	13	17	17	10	18	11	16	2	16	10
Australia	11	11	10	11	1	1	9	7	5	4	4	8	9	2	4	8	12
Malta	3	2	1	3		1	1	2	1	3		2	1	1		1	1
Total	219	204	143	103	67	46	127	102	116	165	164	171	151	104	89	180	149

Table 11: Frequency of teachers with other core subject area teaching expertise by country

Country	English, Literacy	Math, Numeracy	Computer Science	ICT	Design and Technology	The Arts	Languages	Sciences	Physical Education & Health	All areas
US	12	47	85	30	31	5	4	15	4	15
England		1	46	38		1	3	2		3
Italy	1	5	13	7	2	1	2	3		2
Ireland		9	13	11				2	1	5
Scotland		1	18	15	1					2
Australia	2	3	3	6	2	1	1	3		10
Malta			2	4						2
Total	15	66	180	111	36	8	10	25	5	39

Table 12 summarizes computing resources used across the working group countries

Table 12. Computing resources, pedagogies and technologies used across the working group authors' countries

<ul style="list-style-type: none"> • Competitions (e.g. Bebras, FIRST) • Resources provided through Clubs/Outreach programs (e.g. Code Clubs) • Online teacher resource sites for CS (e.g. CAS Computing, Digital Technologies Hub) • Videos • Physical computing devices (e.g. Beebot, Arduino) • Online programming tutorials (e.g. code.org, CS First) • Visualisation tools (e.g. Python Tutor) • Question banks (e.g. Quantum) • "CS Unplugged" resources or CS puzzles • Online visual programming environments (e.g. Scratch, Blockly, Alice3D) • App-based programming environments (e.g. ScratchJR, Kodu) • Online general purpose programming environments (e.g. Python, Javascript) • CS Textbooks (e.g. how to teach/learn a programming language) • General software (e.g. Word Processors, Spreadsheets) • General websites that cover CS topics (e.g. ABC Kids, Commonsense.org) 	<ul style="list-style-type: none"> • Tangible robotic device (BeeBot, KIBO or similar) • Robotic device with app (e.g. Dash and Dot, Sphero) • Makey Makey (or similar) • Programmable card, such as Arduino, Micro:bit • Small single-board computers, such as RaspberryPi • Little bits, or other pluggable sensor kits • Sensors • LEGO Mindstorms or similar • Virtual Reality • Artificial Intelligence/Machine Learning Kits (e.g. Google AIY Vision Kit) • Drones • Other
<p>Artificial Intelligence, Machine Learning resources (Google AIY, Google Experiments)</p> <p>Apps for CS (ScratchJR, Kodu)</p> <p>Computers (Laptop, PC)</p> <p>Drones</p> <p>Online programming sites (Code.org, CS First)</p> <p>Programmable cards & single-board computers (Micro:bit, Raspberry Pi)</p> <p>Puzzles for CS (CS Unplugged)</p> <p>Question Banks (Quantum, Bebras)</p> <p>Robotics (BeeBots, Dash and Dot)</p> <p>Smartphones/Tablets</p> <p>Textbooks for CS</p> <p>Engineering Tools/Kits (LittleBits, 3D printers)</p> <p>Virtual Reality Devices</p> <p>Visualisation tools (e.g. Python Tutor)</p>	

<ul style="list-style-type: none"> • Modelling (incl. live coding, simulation, highlighting, demo, code walkthrough or review) • Targeted programming tasks (e.g. debugging, sabotage, reading and tracing code, fill-in-the-gaps, annotation, Parson's Problems, worked examples) • Socially relevant computing projects • PRIMM • Using narratives, case studies or story • Questioning techniques • Unplugged learning (e.g. embodiment, acting out, sequencing cards) • Studio-based learning (the design and development of a creative product of any kind) • Paired programming • Algorithm design and representation (e.g. flowcharts, tactile sorting) • Project planning and management (e.g. time management, use of GitHub, etc) • Embedding Computational Thinking skills and processes (e.g. decomposition, abstraction) 	<ul style="list-style-type: none"> • Flipped Learning/Classroom • Cross-curricular integration • Mastery Learning • Inquiry-based learning • Project-based learning (including Maker learning) • Work-integrated learning • Direct Instruction • Semantic waves (cumulative knowledge building) • Collaborative learning
<ul style="list-style-type: none"> • Flipped Learning/Classroom • Cross-curricular integration • Mastery Learning • Inquiry-based learning • Project-based learning (including Maker learning) • Work-integrated learning • Direct Instruction • Semantic waves (cumulative knowledge building) • Collaborative learning 	<p>Programming related</p> <ul style="list-style-type: none"> • Targeted programming tasks (e.g. debugging, sabotage, reading and tracing code, fill-in-the-gaps, annotation, Parson's Problems, worked examples) • Unplugged learning (e.g. embodiment, acting out, sequencing cards) • Algorithm design and representation (e.g. flowcharts, tactile sorting) • Project planning and management (e.g. time management, use of GitHub, etc) • Teaching of Computational Thinking skills and processes (e.g. decomposition, abstraction) - covered elsewhere twice now
<ul style="list-style-type: none"> • Flipped Learning/Classroom • Cross-curricular integration • Mastery Learning • Inquiry-based learning • Project-based learning (including Maker learning) • Work-integrated learning • Socially relevant computing • Studio-based learning (the design and development of a creative product of any kind) 	

2.5 Discussion and lessons learned

The final sample of teachers who completed all survey questions was sufficient for a pilot study but limited. The completed pilot survey was represented the experience of 244 teachers

in 7 countries that primarily use English as a native language. The majority, 115 teachers, came from 27 US states. Only 14 teachers completed the survey from Australia and only 6 teachers from Malta. As we offer the survey to additional countries, we will need to reevaluate the item data to ensure the English matches local phrases and terminology for common definitions. If the survey is translated into other languages, it will need to be re-validated (Apolone & Mosconi, 1998). In this section we discuss some key pilot study observations, share our lessons learned during the project and suggest recommendations for future use of the teacher survey instrument.

2.5.1 Pilot Study Curriculum Observations

Our presentation of pilot results provides a sample demonstration of the insights that can be gathered from the country reports and teacher survey instruments. Although working with a small sample size and aggregated results across year bands for countries, we are able to provide early observations around aspects on intended and enacted curriculum descriptives. In terms of intended curriculum, our results demonstrate how contextual information gathered through the country report can assist in making comparisons of schooling contexts and CS curriculum requirements. We observe three broad type of CS curriculum implementation that includes those with a state plan for CS in place, those with no state plan for CS in place and those whose CS state plan is in development. Additionally, we observe interesting patterns across countries for curriculum requirements, such as CS topics and programming languages, noting that some countries have defined programming languages in their intended curriculum that are to be taught at specific year levels and others have not.

In terms of enacted curriculum we observe what CS topics teachers are implementing in the classroom, irrespective of their intended curriculum. We identified that programming skills and concepts, and algorithms were the most common CS content being taught, with machine learning and artificial intelligence being less popular, reflecting the intended curriculum. As (Larke, 2019) mentions, teachers are the gatekeeper to CS education as they choose to interpret and/or reject the intended curriculum. As CS becomes more mainstream in schools and teachers move through phases of curriculum implementation, it will be interesting to determine if enacted CS content diversifies and whether content taught more closely aligns with intended curricula.

The survey also captures information about the equipment and resources teachers use in the classroom for CS and their perceived needs. Interestingly, the results highlight differences between countries, demonstrating the value of such comparisons and that a onesize-fits-all approach may not work for making recommendations about how to support teachers with CS curriculum. For example, differences emerged in relation to teachers requesting CS-specific technology in England and Scotland more than any other resource, compared to Malta who identified needing non-CS specific technology.

This report has highlighted some of the early observations via descriptive reporting. However, the power of the instruments will be realised in future analyses where we can align findings from the intended and enacted curriculum, as has been demonstrated in a recent publication of the work comparing teachers' implementation of programming languages and CS topics in comparison to intended curriculum using the METRECC instrument (Falkner, Sentance, Vivian, et al., 2019a). Additionally, exploring differences in enacted curriculum according to year level bands, in alignment with intended curriculum, will help strengthen our understanding of what is happening in different classroom contexts and required support for primary and secondary teachers. The METRECC instrument not only captures country level reports of intended curriculum, but also enacted curriculum directly from teachers. Up until now enacted curriculum surveys have largely focused on perceptions of what is happening in classrooms (Sentance & Thota, 2013b), (Schrire & Levy, 2012) or narrow areas of CS such as CT (Mannila et al., 2014b).

2.5.2 Lessons Learned

The analysis of the survey, and the pre-processing of the data highlighted several areas that could be improved or what the group found as successful approaches. This may be of value, when considering future survey tools and the processing of the data set, opposed to the survey instrument itself. This section aims to highlight points that may be of value to the CS education community when considering an international benchmark study/survey, perhaps aimed at K-12. While this may not be applicable in all cases, this Working Group feel that they are of value to highlight, and are in order of appearance and not importance.

9.2.1 Survey Testing/Local Pilot. While the survey was tested, one finding was that some jurisdictions struggled with one or two questions, where the Working Group representative reviewed the survey. Perhaps this may have been a minority, but perhaps a subtest or pilot survey per jurisdiction would have proven useful.

9.2.2 Ethics Approval. The Working Group leaders sought and were granted ethical approval or met requirements to conduct the survey in their jurisdictions. The Working Group recommend that an early investigation be conducted to determine if the process could be coordinated across multiple jurisdictions (if at all possible), and suggest that this would be worth preliminary investigation prior to the individual effort.

9.2.3 Working Group Collaboration. The collaborative approach taken worked very well for this Working Group. During the initial survey design, any question that was validated from another study was added to a referencing repository and collaborative documents (in this case we used Mendeley and GSuite). This expedited the work once the group met in person. Additionally, the use of a real-time collaboration LaTeX environment (in this case, Overleaf) enabled members to easily and concurrently work on the report. The referencing repository

also linked into the real time collaboration tool, allowing real time updates of the the bibliography.

9.2.4 Closing the Survey Instrument. When the Working Group closed the survey, the tool used to collect the data, allowed participants to continue. This is not an issue, except when comparing numbers of participants from our downloaded data and the data in the tool, they differed. Consideration of a cut-off and data download time may be of value when working across multiple contexts and time zones.

9.2.5 A Single Survey. This Working Group developed three instruments as mentioned previously, one for Australia, one for the US and one for International participants. There were very minor differences between the surveys (for example, the landing page and in the case of the US survey compared to the International survey, it had an additional breakdown of state/region). This in itself was not an issue or constraint, however, the tool used produced three separate data-sets. This again was not inherently an issue, but took considerable time to combine, while validating the combined data-sets. A fork or conditional in the survey could have been more efficient. This could be alleviated with a coordinated HREC approval effort. A second, but again minor consideration, was that several participants took a survey that was not their intended survey (from a different region). This perhaps was due to the international profile and reach of some of the Working Group members, where if they promoted the survey, their reach would have included participants from other intended survey jurisdictions. Perhaps if the surveys contained some specific details (which was not the case in this survey), this would have posed more of a problem.

9.2.6 Survey Early Exit. It was noted throughout the data preprocessing that multiple participants exited the survey at varying stages. A number of reasons may have caused this but one being that a survey that takes an hour or more is too long for most people, even when provided the option to only answer compulsory questions. Those who did complete the survey typically completed all questions. Another issue was the way in which the survey collected data, that meant that exiting the survey at any time would exclude the participants' data from being used in the survey collection. This again is a valid method in survey "opt-out" implementation. The tool that the Working Group used saved the data after a minimum of one page was passed. This is one of the main reasons, why the sample size reduced so rapidly. This is not an issue for cleaning the data, but it does highlight that many participants started the survey and exited without completing. This was perhaps a lost opportunity. While the Working Group was able to examine what block the participants exited the survey on, the question that they exited on was not available. For instrument validation, this metric would have had significant value, to focus efforts on particular questions, that may have had a high exit rate. It is acknowledged that there could also have been other reasons why participants exited, such as time constraints, but this data could have had value in investigating this.

An addition (which the tool this working used did not have as a feature or was not easily implemented), which may have to be developed as a bespoke survey instrument, was capturing timing between questions and questions blocks as outlined in Table 2. This may also have had value for identifying questions that require further investigation.

2.5.3 Recommendations for Future Use

The survey instrument can be used across different countries. In order to be able to compare the differences between the prescribed and enacted curriculum, the first step to using the survey instrument would be to complete the country report template as described in section 6.1. The country report is designed to capture the prescribed curriculum in the country where the research will be conducted.

A researcher may choose to use different sections of the survey rather than the complete instrument or apply it to subsets of the K12 population such as secondary school teachers. If this approach is adopted then population validity (see Section 7.2.2) would need to be conducted in order to determine the representation of the targeted sample population to ensure its validity across these groups.

The survey might need to be translated to the native language of the population it will be administered to. Face validity (see section 7.2.1) would need to be conducted by language speaker experts in order to ensure that the translated survey remains valid.

2.5 Teachers' Computer Science self-esteem

The METRECC data were studied in more details deepening the self-esteem of Computer Science teachers (Vivian et al., 2020). Self-concept, broadly speaking, is a person's perception of oneself (Shavelson & Bolus, 1982) and has also been used interchangeably with self-esteem (Trautwein et al., 2006), however, (Heatherton & Wyland, 2003) distinguishes self-concept as related but referring to 'the totality of cognitive beliefs that people have about themselves..., everything that is known about the self' (e.g. race, likes, dislikes, values, appearance descriptions), whereas self-esteem 'is the emotional response that people experience as they contemplate and evaluate different things about themselves' (p.220). Self-concept can be both descriptive and evaluative in nature and is influenced by internal and external factors and experiences. While a global self-concept perspective is a view of oneself generally, domain-specific self-concept or self-esteem reflects a person's self-evaluation regarding a specific domain or ability in academic areas, typically through self-reported measures (Trautwein et al., 2006). Similarly, performance self-esteem is one's sense of general competence and includes intellectual abilities, performance, self-regulatory capacities, confidence, efficacy, and agency (Heatherton & Wyland, 2003). Self-esteem is an attitude about the self and is related to personal beliefs about skills, abilities, social relationships, and future outcomes (Heatherton & Wyland, 2003) (Heatherton). (Shavelson &

Bolus, 1982) present self-concept as a multi-faceted construct inclusive of both academic self-concept (subject areas) and non-academic self-concept (peers, significant others, emotional states, physical ability and physical appearance). (Coopersmith, 1967) self-esteem involves an attitude of approval or disapproval and "indicates the extent to which the individual believes himself to be capable, significant, successful, and worth. In short, self-esteem is a personal judgment of worthiness that is expressed in the attitudes the individual holds toward himself" (pp. 4-5).

In our study, we acknowledged that self-esteem and self-efficacy are closely linked but we recognized their differences when measuring these constructs. We took the view that self-esteem is more broadly concerned with a person's positive and negative attitudes or perceptions about their self (Rosenberg, 2015), (Winch, 1965) and within particular domains (domain specific self-esteem/self-concept) (Trautwein et al., 2006). Self-esteem is more concerned with an individual's perception of their self-worth, satisfaction with themselves and their comparisons to peers. On the other hand, self-efficacy is concerned with a person's belief in their own capabilities to execute specific tasks (Bandura, 1997). As an example of differences across global self-esteem, domain specific self-esteem and self-efficacy we present statements that represent variations in the constructs below:

- 1) I feel that I have a number of good qualities (global self-esteem).
- 2) I feel that I have a number of good [Computer Science or programming] qualities (domain specific self-esteem).
- 3) I can write syntactically correct programming statements (self-efficacy).

(Rosenberg, 2015) defines high self-esteem as being at a satisfactory level 'good enough' and describes a person with high self-esteem as having self-worth and respect for themselves. Low self-esteem is aligned with a person feeling rejection in themselves, dissatisfaction and low self-worth. Self-efficacy is concerned with people's perceived beliefs in their capabilities to produce given attainments (Bandura, 1997). Self-efficacy originates from the work of Albert Bandura (Bandura, 1997) who defines self-efficacy as "people's beliefs about their capabilities to produce designated levels of performance that exercise influence over events that affect their lives". (Bandura & Wessels, 1994). It is a central powerful belief about one's capability to accomplish and perform a task and is a cognitive function that supports individual's behavior.

In the work based on the METRECC survey (Vivian et al., 2020) we used publicly available data (n=219) from a pilot study using a Teacher CS Self-Esteem scale. Analysis revealed significant differences, including:

- 1) females reported significantly lower CS self-esteem than males,
- 2) primary teachers reported lower levels of CS self-esteem than secondary teachers,
- 3) those with no CS teaching experience reported significantly lower CS self-esteem,
- 4) teachers with 0-3 years experience had a negative CS self-esteem, but after four years, teachers had a positive CS self-esteem,

- 5) teachers who lived further from metropolitan areas and in some countries reported lower CS self-esteem.

2.6 International comparison of Intended and enacted curricula

In a successive work (Falkner, Sentance, Vivian, et al., 2019b) we presented an international study of K-12 Computer Science implementation across Australia, England, Ireland, Italy, Malta, Scotland and the United States. We present findings from a pilot study, comparing CS curriculum requirements (intended curriculum) captured through country reports, with what surveyed teachers (n=244) identify as enacting in their classroom (the enacted curriculum). We address the extent that teachers are implementing the intended curriculum as enacted curriculum, exploring specifically country differences in terms of programming languages and CS topics implemented. Enacted curriculum in classrooms should reflect the curriculum policies of the state (the intended curriculum) (A. C. Porter & Smithson, 2001b). This led us to interrogate our data from the METRECC survey (Falkner, Sentance, Vivian, et al., 2019d) to investigate the following research questions:

- 1) What are the similarities and differences across countries in terms of intended CS curriculum topics and programming requirements?
- 2) To what extent are teachers addressing the intended CS curriculum with their enacted curriculum in classrooms?

Figure 2.1 shows a Comparison across countries for programming languages implemented across age groups. The study presented pilot results from 244 participants across seven countries. We have focused this paper on the alignment between intended and enacted curriculum in the areas of topics taught and programming languages used. We see these as critical areas for further analysis and monitoring not only in terms of alignment and its ensuing benefits, but also in relation to our assumptions as tertiary educators on prerequisite knowledge and experience. We have identified that both visual and text-based programming languages are being used across K-12 by some teachers, warranting further research into potential impact on student learning and motivations. We also identify that unplugged activities are commonly used across K-12, extending into senior years despite this not being explicitly defined in intended curricula. Furthermore, we notice teachers' motivations for programming language choice is consistent across countries. Interestingly we expected that curriculum would drive teachers' motivations for selecting programming languages, however, our results discovered this isn't the case and that student-driven factors motivate selection. A limitation of this study is that results are based on a small pilot sample size, particularly for some countries, however, future work will seek to survey a larger sample across multiple countries. Nevertheless, this study demonstrates the value in investigating intended versus enacted curricula in terms of recording teachers' curriculum enactment and in identifying differences with curriculum alignment across countries. These insights can potentially be used to guide further curriculum reform, or the development of targeted resources and/or

professional development to better support teachers in implementing and delivering new CS curricula. This work has focused exclusively on the aspects of CS topics and programming languages implemented, however, there are opportunities to explore other forms of enacted curriculum such as CS resources used and pedagogy. A further limitation is that we have focused our analysis and presentation of results on K-12 broadly, with future research warranting a breakdown of analysis into primary and secondary years to determine if there are differences, as well as other factors that impact on programming language and CS topic implementation.

Country	Ages	Un plugged	Symbolic (no text)	Visual (text)	Hybrid	Text based
Australia	3-5	21%	21%	21%	14%	7%
	6-7	43%	43%	43%	14%	14%
	8-10	64%	64%	64%	14%	29%
	11-12	79%	79%	86%	21%	43%
	13-15	14%	7%	21%	0%	14%
	16-17	7%	7%	7%	0%	7%
	18-19	0%	0%	0%	0%	0%
England	3-5	8%	6%	10%	4%	10%
	6-7	17%	15%	17%	6%	13%
	8-10	21%	17%	21%	6%	13%
	11-12	60%	40%	54%	10%	58%
	13-15	65%	37%	58%	10%	73%
	16-17	63%	31%	52%	8%	67%
	18-19	48%	21%	33%	2%	48%
Ireland	3-5	0%	0%	0%	0%	0%
	6-7	5%	4%	4%	0%	0%
	8-10	16%	6%	8%	2%	11%
	11-12	21%	8%	12%	2%	21%
	13-15	47%	10%	23%	2%	68%
	16-17	47%	12%	23%	2%	68%
	18-19	26%	10%	15%	0%	42%
Italy	3-5	0%	0%	5%	0%	0%
	6-7	5%	5%	10%	0%	0%
	8-10	15%	15%	20%	5%	10%
	11-12	10%	10%	10%	0%	10%
	13-15	30%	35%	40%	10%	40%
	16-17	45%	55%	55%	20%	65%
	18-19	40%	50%	50%	15%	60%
Malta	3-5	0%	0%	0%	0%	0%
	6-7	0%	17%	0%	0%	0%
	8-10	0%	0%	0%	0%	0%
	11-12	17%	17%	33%	17%	17%
	13-15	33%	33%	17%	33%	33%
	16-17	17%	17%	0%	17%	17%
	18-19	0%	17%	17%	0%	0%
Scotland	3-5	6%	6%	0%	0%	0%
	6-7	6%	6%	6%	0%	6%
	8-10	11%	6%	11%	0%	11%
	11-12	44%	17%	50%	11%	50%
	13-15	83%	50%	89%	17%	89%
	16-17	78%	50%	83%	17%	83%
	18-19	33%	22%	33%	11%	33%
USA	3-5	3%	3%	3%	1%	2%
	6-7	10%	8%	9%	2%	5%
	8-10	10%	9%	10%	2%	4%
	11-12	28%	16%	28%	10%	19%
	13-15	60%	31%	61%	23%	51%
	16-17	56%	27%	55%	20%	51%

Figure 2.1 A comparison across countries for programming languages implemented across age groups.

We recommend ongoing research to continue to survey and monitor the landscape to determine whether enacted curriculum implementation changes over time as intended curriculum implementation in countries matures.

Chapter 3: Competencies

Objective of the chapter: Codifying the knowledge and skills areas of the computer science 2013 curriculum into a format consistent with the CC2020 project competency framework.

Approach: an input-process-output model. The inputs were the CS2013 curriculum model and CC2020 conception for modeling an effective competency statement. The outputs would be the report indicating how the creation of competency statements operated and how they could be developed, complemented by a transformation of the competency statements into a format that would readily feed into the CC2020 prototype visualization tool, to serve as exemplars of competencies to be visualised by this prototype

Result achieved and novelty: This chapter demonstrates one way in which the transition from current learning-outcomes-based practices to the competency-based practices can be approached.

Significance for the state of the art and the narrative of the work: The chapter discusses the challenges and insights that have emerged as the learning outcomes for various Knowledge Areas in the CS2013 report were re-expressed in terms of competencies.

In a successive working group in 2020 (Clear et al., 2020a) we analyzed the broadly influential document Computing Curricula 2005 (CC2005) comparing it with the Computing Curricula 2020 (CC2020). CC2020 provides a vision for the future of computing education, including a comprehensive report that contrasts curricular guidelines, and contextualizing those guidelines within the broader landscape of computing education. In the process, a framework of competency-based educational principles has been developed which is closely aligned with other skills and qualifications frameworks. The working group report (Clear et al., 2020a) demonstrated one way in which the transition from current learning-outcomes-based practices to the competency-based practices can be approached.

The process adopted by the team evolved as the work progressed. Broadly speaking, we had conceived the approach as adopting an input-process-output model. The inputs were the CS2013 curriculum model and CC2020 conception for modeling an effective competency statement. The process involved mapping from CS2013 to CC2020 expressions of competency. The outputs would be the report indicating how the creation of competency statements operated and how they could be developed, complemented by a transformation of the competency statements into a format that would readily feed into the CC2020 prototype visualization tool, to serve as exemplars of competencies to be visualised by this prototype.

3.1 From CS2013 to CC2020

The CS2013 Curriculum Report provides curricular guidelines centred around Knowledge Areas (KAs), Knowledge Units (KUs), Topics, and Learning Outcomes (LOs). By contrast, CC2020 specifies competencies as a composition of Knowledge and Skill elements

associated with Disposition elements, all within the context of performing a specific Task. The CS2013 Body of Knowledge provides us with Knowledge elements and associated Skill elements (through the LO levels of mastery). But it does not provide any of the Disposition or Task elements. Thus, mapping individual KAs or KUs to Competency Statements requires us to envision a specific professional context (the Task and Disposition elements) within which the KA/KU is demonstrated. Having established work partners, we set to the task of writing up sample Competency Statements for individual KAs/KUs in CS2013. Each pair worked independently to generate four statements or more (a minimum of two statements for each of two KAs/KUs). Some partners were able to carry out this work synchronously; others used a divide-and-conquer approach: each partner wrote up two statements, which were then reviewed by the second partner.

3.2 Level of Granularity

When we regrouped to review and discuss our draft statements, we quickly identified key differences in each pair's work.

- 1) Approaches ranged from capturing a small number of related topics/LOs in a statement to capturing entire KAs in a single statement or multiple statements; after discussion, we agreed that it was most appropriate to express competencies at the level of entire KAs, from the point of view of a graduate or professional. When we "designed" a competency statement, we focused on a particular knowledge area and looked for collections of topics and/or learning outcomes (LOs) that could be observed in tasks and at particular skill levels. While seemingly straightforward, there was significant variance in how this was approached, and how the resulting competency was formulated. Dispositions were often implied at first, but refinement of the statements made these both more clear and more relevant.
- 2) The format ranged from simple textual lists to more complex tables linking Knowledge-Skill pairs with individual dispositions; after discussion, we agreed on a tabular format for ease of integration with the CC2020 prototype. This also aligns with an anticipated need from practitioners/curriculum designers for an easy-to-use instrument in a tabular format, and precise instructions.
- 3) We discussed writing dispositions based on individual Los vs. dispositions based on observable tasks that encompass several LOs. As per the CoLeaF model, we agreed that dispositions should be defined based on observable tasks.
- 4) We argued that all eleven dispositions are important and expected for a computer science student or professional. However, not all dispositions are equally relevant for a task, and therefore we agreed to include only the most pertinent dispositions to the task at hand.

- 5) We discussed at what level competencies should be written (e.g., module, course, speciality, or program). We concluded that competencies should be written at any level decided by practitioners/curriculum designers.
- 6) We discussed the importance of providing the context in which tasks are carried out, in addition to expressing measurable/ observable achievement.
- 7) Context specification turned out to be highly subject to granularity issues. Over-specifying the context tended to leave the competency statement looking more like a classroom assignment; under-specifying the context/task could easily disrupt the observability of the competency in a way that made it difficult to know whether a particular instantiation of the context/task actually met the intent of the statement.

Appendix B of the published report show samples of the original statements produced by each pair. This demonstrates the wide variation we had to accommodate to come up with a common format.

3.3 Common Format

Both textual and tabular statements are generated by starting from a particular topic/LO, or a group of closely-related topics and Los from one KU or an entire KA, imagining a concrete task where a student or professional would be expected to demonstrate the target topics/LOs, and writing up a clear informal statement of this situation. Samples of these LO and topic statements from CS2013 are provided in Tables 1 and 2 in Section 3 of the report (Clear et al., 2020a). Using the informal statement as a starting point, we then review other topics and LOs (mostly in the same KA but this could also cross KA boundaries) to identify the ones most relevant to the statement and generate a list of Knowledge-Skill pairs. In these pairs, the topic/LO constitute the Knowledge component, the level at which it is involved constitute the Skill component—note that this level may differ from the one indicated in CS2013, most often by being higher up in Bloom’s hierarchy. After all, while a topic in CS2013 might be phrased in terms of students “being aware” of a particular topic, genuine situations where students need to demonstrate that awareness will most often involve some sort of application (and therefore be at least at Bloom’s Application level). Disposition elements are then added for the entire statement; they are not tied to specific Knowledge-Skill pairs because they apply to the overall situation being described. After much discussion, we settled on the tabular format as being the most appropriate: it encapsulates every key element that makes up a competency in a compact form, and it is especially well-suited for automated processing—in particular, by the CC2020 prototype system.

3.4 Validation Methodology

To develop a common language and follow a consistent and robust process when defining our competencies, we used a two-step validation strategy. Following our first attempt at producing draft competency statements, individual drafts were subjected to a thorough

review. First, a rubric was agreed upon (see sub-section 4.4.2) to evaluate the consistency and validity of each statement. Next, each pair of original authors reviewed their statements (self-evaluation), and larger groups were formed from two pairs to validate each other's statements (peervalidation).

3.4.1 Competency Statement Self-Evaluation.

Each pair of original authors reviewed their competencies and re-wrote their statements. Each statement was reviewed to identify issues, undertake revisions, and consider any limitations found for inclusion in the discussion section. The revision process had side-effects besides expressing competencies in a more robust and complete format. In many cases, the authors identified and included more statements. In addition, the process of self-evaluation resulted in the following improvements.

- 1) The most common improvement was to make the dispositions expected in the competency more explicit within the statement. For example, the first drafts of the IS, SE and SP statements used more implicit language to indicate the dispositional elements expected.
- 2) Some authors identified and included more competency statements as a direct result of their revisions. For example, including more LOs missed in the initial pass, or pulling in topics or LOs from other KAs that were clearly required to support the desired competency demonstration. This often included expanding a single-KA statement to include aspects of other KAs.
- 3) Context improvement: we discussed how to express the context when writing competencies that apply to topics taught at different levels and/or using different technologies, such as an HCI course taught using mobile technologies vs. one using web development as the vehicle of instruction. We found that the context should be neither over- nor under-specified, but remain sufficiently open. This has the merit that it allows for future formulations of competencies in a more specific context to a particular program or course, and the drawback that the formulation could be perceived as too general and vague.
- 4) We found that some competencies overlap (e.g., CS-SE-3 parallels CS-SE-1 to a certain extent). This led to a discussion of the way in which some competencies could subsume others. We consider that some amount of overlap is nonetheless acceptable and needed as competencies could map differently depending on the particularities of a program, speciality or course.
- 5) The level of the competency statements and the context in which these statements are employed became a point of discussion. For example, numerous statements seemed closely associated with a particular instantiation of a particular course (e.g., HCI-3, several of the OS statements). This was most obvious as statements

incorporated skill levels less than three. Similarly, the statement could be at a program level, where the skill levels were above, or routinely above, level four.

3.4.2 Competency Statement Peer-Validation.

Within the sub-groups of four, each statement was reviewed by non-author peers. This step aimed to develop team norms within the sub-group about the statements, and to establish a common approach and coverage. It is interesting that the sub-groups report that, even if peer validation resulted in valuable improvement of the competency statements, there were more similarities than differences of opinion within the sub-group. This gave us more confidence that the process we employed is effective in developing a common language. Our peer validation involved developing a rubric around sets of questions that could be assessed for the competency specification, examining both the free-form statement and the mapping to its constituent K-S-D components.

- 1) Comprehensible & Clear? {Yes | Maybe | No}
- 2) Context provided? {Yes | Implied | No}
- 3) Observable achievement? {Yes | Maybe | No}
- 4) Embodies Dispositions? {Yes | Implied | No}
- 5) Incorporates Professional KAs? {Yes | Implied | No}
- 6) Good Coverage of KA/KU Content? {Yes | Maybe | No}
- 7) Effective mapping of statement to connected K+S+D? {Yes | Maybe | No}
- 8) Level of CS2013 Competency Statement {Program | Multiple KA | Single KA}
- 9) KU Count {number}
- 10) Topic Count {number}
- 11) LO Count {number}

Other information collected from the review included Notes specific to this statement, Who conducted the review and when, General thoughts about the authoring process and Suggested rewrite information. Some, but not all statements were rewritten following review. Figure 1 shows the tabular format for competency specification.

In the following we report the validated competencies statement for algorithms and complexity.

AL—Algorithms and Complexity.

- 1) AL-1 Given a problem to program for which an inefficient solution is not sufficient, research and implement a solution that is sufficient for reasonable instances, without straining other resources or being too complex to implement.
- 2) AL-2 Given a problem to program for which there may be several algorithmic approaches, evaluate them and determine which are feasible, and select one that is optimal in implementation and run-time behavior.

- 3) AL-3 Given a problem that involves multiple actions (such as insertions, deletions, and searches) that may or may not have predictable occurrences, properties, or relationships, devise a data structure that optimally supports the actions.
- 4) AL-4 Present to an audience of co-workers and managers the impossibility of providing them a program that checks all other programs, including some seemingly simple ones, for infinite loops.
- 5) AL-5 Given partial specifications for a program that is to include a small user-interface language, devise a syntax that accommodates the expressiveness of the interface, is simple to use and not prone to user mistakes, and can be implemented using a small finite state machine solution. Present the syntax to the users as regular expressions
- 6) AL-6 Prepare a presentation that introduces first year students to the basic concepts of algorithmic complexity, such as notation, characteristics, complexity classes, time and space, empirical measurement, and impact on practical problems.
- 7) AL-7 After watching algorithms' animations, provide in natural language, different descriptions, starting from the high-level idea and the algorithmic strategy down to details covering implementation, execution time, and the underlying mathematical model.
- 8) AL-8 In different context from pre-university to undergraduate and graduate level up to job interviews, apply algorithm design strategies to solve an assigned puzzle and prove the correctness and eventually the optimality of your solutions.

The developed competencies statement for discrete structures were:

DS—Discrete Structures.

- 1) DS-1 Given a problem involving analysis of populations (such as traits, geographic origins, social status, etc.), design a solution that utilizes sets and set operations in the processing and presenting of results in response to user queries.
- 2) DS-2 Given a problem, which in the general case has no efficient solution, and a proposed efficient solution for a contraction of the problem, prove that the solution is correct.

KA-Ref#	Title	Competency Statement	Dispositions	Knowledge-Skill Pairs			Traced Competencies	Notes
				KU / Broader Knowledge Description	Topic/LO Description	Knowledge Element		
Our reference number goes here. E.g., "KA-###"	Short Title	Competency statement goes here (Natural Language) that embeds the task/context and suggests the K-S-D breadth and depth	List top applicable dispositions: <ID+descriptor>. Note that dispositions are not aligned with KS-pairs.	CS2013 or other description of a KU	Text of LO or Topic or Professional Skill	CS2013 LO or Topic ID	Bloom's Level	Any notes or comments
					Text of LO or Topic or Professional Skill	CS2013 LO or Topic ID	Bloom's Level	Often null
					Text of LO or Topic or Professional Skill	CS2013 LO or Topic ID	Bloom's Level	
				CS2013 or other description of a KU	Text of LO or Topic or Professional Skill	New IDs for extensions to CS2013	Bloom's Level	Note deviations from Bloom's level in CS2013
				Text of LO or Topic or Professional Skill	CS2013 LO or Topic ID	Bloom's Level		

Figure 1: Competency specification in a tabular format

3.5. Preliminary Analysis

Initially, 32 sample competency statements were developed for eleven KAs (AL, AR, DS, HCI, IS, NS, OS, PL, SDF, SE and SP).

Further reviews, quality assessment and statement development continued for eight of these KAs, as well as the inclusion of the CC2020 Professional & Foundational Knowledge elements outlined in Table 4 or the working group report (Clear et al., 2020a). From the second round of statement analysis and refinement, 43 competency statements were developed addressing the AL, AR, HCI, IS, NC, OS, PL, SDF, SE and SP CS2013 KAs and the CC2020-PK knowledge area. Table 11 of the report (Clear et al., 2020a) depicts how these ten KAs were 'covered' by the 43 competency statements authored. Text and summary information for the original competency statements drafted for the 11 KAs is presented in Appendix C of the work (Clear et al., 2020a). For the 43 competency specifications summarized in Table 11 of the report (Clear et al., 2020a), the KU coverage was computed as the sum of the Tier-1 and Tier-2 topics and Learning Outcomes for that KA, divided by the sum of the Tier-1 and Tier-2 topics and LOs that were addressed by one or more competency statements for that KA. The Topic/LO. Coverage is shown as that ratio expressed as a percentage (%). The last column of Table 11 shows the ratio of topics & LOs per statement, which provides an overview of the complexity of the competency statements for that KA. Lower ratios indicate that the competency statement focused on fewer CS2013 Topics/LOs. Higher ratios indicate that the statement has higher/broader expectations and or a higher expectation for the synthesis of knowledge in the demonstration of that competency. As Table 11 suggests, the competency statements developed for Programming Languages (PL) did not include tracing to the related topics; this shows that a different level of granularity was used by these authors who only specified LOs. This differed from authors of the other KAs who traced both topics and LOs. Consequently the coverage for these statements appears significantly lower (26%) than most other KAs.

As a whole, these competency statements represent ten of eighteen KAs, and 50% of the Tier-1 and Tier-2 topics defined in the CS2013 for these ten KAs. At a glance, these 43 statements represent over 300 Topics and LOs, or roughly a seven-fold reduction in the amount of information conveyed. The notion of coverage, and the relationship of competency

statements to each other and to the underlying knowledge and skills expected of a program are points of expansion that we leave to future work.

3.6. Discussion

Formulating competencies which embody the expected learning outcomes (LOs) of earlier ACM/IEEE curricula requires a combination of interpretation and expert judgement. Our point of departure as a working group was to explore how the CS2013 curriculum, in combination with our professional and disciplinary insight, might be amenable to re-expression. The goal was to transform the knowledge-and-learning-outcomes-based CS2013 curriculum into formulations of expectations of graduate proficiency expressed in terms of competencies. One key output was, thus, to develop a set of steps (a process) illustrating how existing curricula might be recast in terms of a learning-outcomes based collection of competencies. This section offers these steps and summarises the lessons learned in that process. It seems likely that similar efforts in other fields and sub-fields of computing will directly benefit from these insights. The working group hopes that the processes and quality assurance rubrics developed in this project will serve the profession well, as communities of educators in the computing sub-disciplines engage in related exercises in the future.

3.6.1 Steps for Re-expressing LO-based Curricula in terms of Competencies

Our recommendation, based on the work described here, is that the following general process provides good support for re-expressing LO-based approaches in terms of competencies.

- 1) Start by identifying the knowledge areas. Often these would be identified during the development of the LO-based approach. It is best to use generally accepted vocabulary, such as that used in the CS2013 report.
- 2) Use the LO itself to identify the skill level. See the discussion below.
- 3) Developing the right set of dispositions is most easily accomplished by drawing on the expert judgement capacity of colleagues (for instance at curriculum committee or board of studies), whereby the overall curriculum intent and the knowledge area plus the learning outcome can be used to identify a relevant set of dispositions.
- 4) Identify a task to frame the competency statement. For a guidelines report like CS2013, the tasks can quite often be "verbs that describe professional activity". Some other techniques are discussed in the earlier parts of the paper.
- 5) Develop rubrics to ensure good quality. Use them to review the resulting competency statements and validate them in terms of process and knowledge unit completeness. The work of the ITiCSE working group has explored a variety of mechanisms to execute these steps, and has also taken a self-critical perspective. We offer some concrete suggestions for how to carry out the broad steps above. See, for example, Section 4.3 of the report (Clear et al., 2020a) for the details of the common format we

developed for writing competency statements, and Section 4.4.2 (Clear et al., 2020a) for a discussion of our validation approach.

3.7. Examples of competencies statements

The 43 competency statements (not the full specifications) are listed in the subsections that follow. These subsections list each of the free-form text statement of the preliminary competencies developed with that KA in mind. Note that more statements were developed; these 43 sample statements were subject to quality review, and many were improved in response to the issues uncovered.

3.7.1 AL—Algorithms and Complexity.

- AL-1 Given a problem to program for which an inefficient solution is not sufficient, research and implement a solution that is sufficient for reasonable instances, without straining other resources or being too complex to implement.
- AL-2 Given a problem to program for which there may be several algorithmic approaches, evaluate them and determine which are feasible, and select one that is optimal in implementation and run-time behavior.
- AL-3 Given a problem that involves multiple actions (such as insertions, deletions, and searches) that may or may not have predictable occurrences, properties, or relationships, devise a data structure that optimally supports the actions.
- AL-4 Present to an audience of co-workers and managers the impossibility of providing them a program that checks all other programs, including some seemingly simple ones, for infinite loops.
- AL-5 Given partial specifications for a program that is to include a small user-interface language, devise a syntax that accommodates the expressiveness of the interface, is simple to use and not prone to user mistakes, and can be implemented using a small finite state machine solution. Present the syntax to the users as regular expressions.
- AL-6 Prepare a presentation that introduces first year students to the basic concepts of algorithmic complexity, such as notation, characteristics, complexity classes, time and space, empirical measurement, and impact on practical problems.
- AL-7 After watching algorithms' animations, provide in natural language, different descriptions, starting from the high-level idea and the algorithmic strategy down to details covering implementation, execution time, and the underlying mathematical model.
- AL-8 In different context from pre-university to undergraduate and graduate level up to job interviews, apply algorithm design strategies to solve an assigned puzzle and prove the correctness and eventually the optimality of your solutions

3.8 What was Learned During the Process?

- 1) Transformation of a traditional curricula document cannot be done mechanically. Competencies capture dispositions explicitly. Learning outcomes do not explicitly address the relevance of dispositions for professionalism and integrity. Developing competency statements thus often requires the introduction of dispositions that may not have been easy to consider, or include, in a learning-outcomes-based approach. Knowledge areas that demand significant systems level work, for example the competency OS-2, require considerable attention to detail, a disposition that needs to be identified and explicitly included when working with our transformative approach.
- 2) Competencies derived from CS2013 are not universally valid, nor necessarily directly appropriate to all computing subfields. Competencies expected of graduates in relation to a particular knowledge area differ and are dependent on degree program objectives. Consider the operating systems knowledge area. One expects a science-oriented graduate to gravitate mainly towards correctness and efficiency of an OS, while an engineering interest would be concerned with efficiency and implementation challenges, and systems support personnel would look for reliable deployment. The curricular statements for different degree programs would steer the same knowledge areas towards their intent by distinguishing the level of the skills to be acquired and their disposition. In this sense competencies provide the ability to further tailor more traditional curricula to the local context of degree programmes.
- 3) Given that competency statements describe the educational value for participating stakeholders, there are two natural views to a given set of competency statements: the curriculum design view and the curriculum consumption view. The design view is concerned with building up the competencies through the curriculum design of an educational program and typically exhibits a locally relevant context that is used to frame competency statements. The consumption view consists of the final competencies available to the stakeholders through an educational program. A guidelines report like the CS2013 might be expected to focus on the consumption view to be globally relevant. Due care must be exercised to ensure that the appropriate sense of context is incorporated into the competency statements.
- 4) It is possible that a stated learning outcome is almost identical to a competency statement. For example, the competency statement OS-6 is almost identical to the learning outcome in CS2013, since the outcome is inextricably linked with connotations of disposition. Interestingly, it also requires the support of a few other learning outcomes from the same knowledge area to formulate a complete competency. While developing a competency statement for this learning outcome, we also observed that it required a higher Bloom's level than that specified in CS2013 in order to capture what we judged to be the intent.

- 5) Formulating competencies also leads to a discussion of the level of abstraction at which a competency could, or should, be formulated. It is not clear that all KS-pairs delineated in a curriculum, such as CS2013, are obligatory when formulating a competency specification. In addressing the level of abstraction at which competencies are formulated we recommend the architects of future curricula to consider if earlier curricula contain KS pairs that may be optional; and where optional KS pairings might exist, upon what criteria they could be included, or excluded, from a given higher level competency. In addition, we recommend that the issue of whether optional pairings can/should be included in a competency be explicitly addressed. Developing complete, observable, contextualized competency statements around skill level B-II turned out to be difficult, even to the point where we briefly considered extending the dispositions of Table 6 to include the concept of "Articulate". This proposal was discussed and rejected because the term 'Articulate' is more of a quality statement about the application of professional knowledge than a proper disposition (Frezza et al., 2020). In particular, the concept of 'articulate' related to B-II topics and LOs is more correctly related to the demonstration of professional communication (e.g., PK-1, PK-2 in Table 4 of the working group report (Clear et al., 2020a)).
- 6) One of the most important insights gained in our transformation exercise was the identification of what we consider to be some missing LOs and topics in CS2013. Therefore, it is essential to divulge that CS2013 should not be seen to represent the sole input for this work. Rather, our work includes additional disciplinary expertise and captures new insights that should inform future revision of CS guidelines, in particular with respect to completion of missing LOs and/or topics. In particular, the absence of professional knowledge areas, such as problem-solving, relationship management, etc., or professional attributes such as leadership. In addition to this challenge, many competencies assume knowledge of areas such as inter-cultural communication, technical writing and rhetoric, which in CC2020 have been termed foundational and professional knowledge areas (see the forthcoming CC2020 report, Table 4.2 or the draft provided in Table 4 of the report (Clear et al., 2020a)). These foundational and professional knowledge areas undergird many competencies and are needed to augment the CS2013 document in order to help to define competencies that cannot be well formulated otherwise. This was an issue specifically identified in the AR-1 example presented in Section 4 of the report (Clear et al., 2020a). Our observations regarding Professional Knowledge are important at many levels. The issues related to foundational and professional knowledge areas extend well beyond more effective competency specification. The broad KAs of Table 4 (Clear et al., 2020a) are really much more like other 'cross-cutting' KAs described in the SWECOM (I. C. Society, 2014). These KAs represent a relatively complete collection of what are colloquially (and incorrectly) referred to as 'soft skills'. The need for CS students to develop skill

and disposition in these KAs is a well-identified problem in computing education (Billett, 2009), (Eraut, 2010), (Frezza et al., 2019), (Nylén et al., 2017), (Waguespack et al., 2019). This appears to be an area where a future version of the CS2013 could be improved. The CS2013 document, though immensely useful, demonstrates a paucity of discussion of what foundational and professional knowledge is required of computer science graduates, with only professional communication and some aspects of teamwork having been included. (See the related topics and LOs in the SE-SPM Software Process Management and SP-PC Professional Communication KAs in CS2013). This work suggests that how to frame this language dealing with professional and foundational knowledge needs more attention as we move towards a competency-based approach to CS education.

- 7) The level of mastery at which LOs are written is also linked to levels of abstraction, and ultimately to the issue of whether a competency can be assessed. Familiarity-level LOs are needed; in fact they are prerequisites of the LOs at the usage and assessment level. In our work, we found the need to re-write some LOs as we found that student should be above the mastery level specified by CS2013. These were needed so they could connect to the task and be observable. For example, we extended the learning outcome AR-ALMO-4 ‘Summarize how instructions are represented at both the machine level and in the context of a symbolic assembler’ written at the Bloom’s taxonomy level II with an enhanced LO ‘Compare and contrast how instructions are represented at different levels of representation starting at the machine level through to a higher level representation’ that is written at a higher Bloom’s taxonomy level (IV).
- 8) Unsurprisingly, some LOs and topics span a range of aspects of professional competence, and can be included in more than one competency. This may appear counter intuitive, but is actually desirable, as students can be exposed to (or use) the same concepts and/or learning material in different modules or courses. In our competency schema, we aimed to meet the expected Tier-1 and Tier-2 LOs as a baseline. By doing this, we believe that we can contribute best to the community through presenting this competency modeling view of CS2013. Practitioners and curriculum designers should find guidance and inspiration in this report which will help them to write their own competencies (e.g., SE statements in Section C.1.16).
- 9) Combining a top-down and bottom-up approach to competency statement development was sometimes difficult. It was far easier to develop statements that were aimed at a graduate—using higher Bloom’s levels, more dispositional terminology, etc. However, these were often more difficult to connect to the LOs and topics of CS2013. It was particularly difficult to connect well-written program-level competency statements (e.g., NC-1 in Section C.1.10).

10) As Table 11 in (Clear et al., 2020a) implies, we have only mapped a subset of CS2013 KUs. One of the disadvantages of working on a sub-set of the CS2013 is that identifying a clear hierarchical structure between competency statements has not been possible. Such a structure might potentially have allowed us to write smaller (or at the very least, more focused) competencies with clear dependencies between them. As it is, we end up with some competency statements that are slightly bloated because we are forced to include a number of “prerequisite” K-S pairs that might be better expressed through dependencies among competencies.

11) We discovered that the “purpose-driven” disposition seemed applicable for several competencies, but at a higher Bloom’s taxonomy level—at level IV or V. How to avoid adding all of the dispositions to most or all of the statements was occasionally difficult. We concluded that retaining an emphasis on the few key dispositions, considered most applicable for the given competency, was the most viable approach.

This work has explored a number of aspects of competency writing. In some stages of the process we adopted a curriculum-developer’s perspective. The latter had to be introduced in an otherwise curriculum developer agnostic document like CS2013 to augment the LOs with the additional information that competencies demand. These processes, lessons-learned, rubrics and coverage metrics all could be used to develop authoring tools for generating/recording competency statements that are part of computing curricula. Similarly, the formats employed in this work, the connecting free-form text to linked knowledge-areas, topics and learning outcomes from formal curricular documents (like CS2013) may well support the visualization development efforts being explored for comparing computing programs using competencies (Waguespack & Babb, 2019).

In his recent work inside the Computer Science Teachers Association Professional Development committee⁴⁶ the author of this work as suggested to use the visualization tool⁴⁷ described in (Clear et al., 2022) to map student and teachers competencies in K-12 context (CSTA, 2017d), (CSTA, 2020b), (CSTA, 2016) to teachers’ professional development curricula as well as computing curricula for k-12 students.

⁴⁶ <https://csteachers.org/page/quality-pd>

⁴⁷ <https://cc.spc.org.pe/>

Chapter 4: Learning resource development and assessment

Objective of the chapter: address the design, development, and assessment of inclusive and accessible learning resources.

Approach: By presenting both the process and the product it is intended to offer a proof of concept for the importance of: 1) the research on the educational practice; 2) computer science education as a per se research domain

Result achieved and novelty: This chapter presents the design, development and assessment of computing curricula suitable for high school students from grade 9 to grade 13, undergraduate students, and teachers both pre and in service professional development.

Significance for the state of the art and the narrative of the work: A method and a proof of concepts for disci, multi, inter and transdisciplinary computing Curricula Development

This chapter will address the design, development, and assessment of inclusive and accessible learning resources. By presenting both the process and the product it is intended to offer a proof of concept for the importance of:

- 1) the research on the educational practice
- 2) computer science education as a per se research domain.

This process has been supported, besides the review of the state of the art presented in chapter 1, by the spiral research path (Caspersen, 2022). Figure 1 shows the first and last milestons of a more than a decade learning path. The whole learning path was presented at presented as part of the keynote introductory speech at the third Italian Scientix conference.

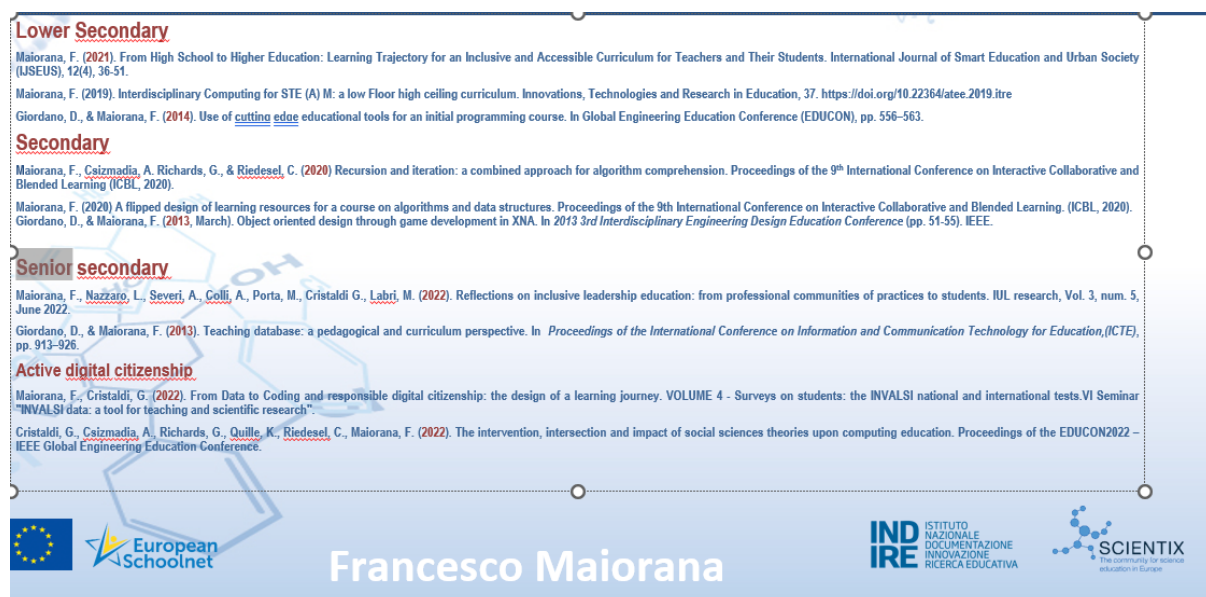


Figure 1: research spiral path supporting the computing curriculum design process.

In designing and developing and assessing the learning resource the following key observations have been distilled from the previously presented studies to guide the work:

- 1) According to the METRECC instrument and the country reports described in chapter 2 and the data analysis it is emerged the importance to grant flexibility to teachers and educators in implementing intended curricula. To cope with this need, coupled with the even greater flexibility required by students, when designing learning resources, it is imperative to have a great variety at all levels: content, learning paths, type of media used in delivering the learning resources, type of assessment, level of deepening, formative and summative assessment, technologies, etc...
- 2) The underlying research-based international comparison of intended and enacted curricula for each of this grade band can inspire, support, and sustain the learning resource design, development, and assessment across near grade band, i.e., for this work, the first two years of high school, final years of high school, undergraduate studies in this work. This chapter is intended as proof of concepts in this direction.
- 3) The comparison of successive editions of computing curricula and the process used for this comparison can be applied in other contexts, e.g., in the relation to the previous point. Teacher research-based self-reflections by comparing and contrasting different editions of their enacted curricula can be another application domain and sustain the spiral professional development process. Again, this chapter is intended as proof of concepts in this direction.
- 4) The competencies-building process (Clear, 2020) performed through the curriculum design process must guide the learning resource design, development, and assessment enacting the identified competencies-building process.
- 5) The frequencies of teachers reported content taught by countries reported in chapter 2 and in the METRECC work (Falkner, 2019), has supported and provided evidence for the choice of content domain: the domain most frequently taught and the reported need for learning resource provide evidence of the potential impact on the teachers, the educators, and their students.

In times of crisis like the one we are currently experiencing (pandemic and war), it is of paramount importance to be highly focused on quality teaching, finding ways to reach everyone. In this chapter I will talk about all the aspects of the Technological Pedagogical Content Knowledge framework (Maiorana et al., 2017b), (Maiorana, Richards, Lucarelli, Berry, et al., 2019a). Success in all three aspects of the framework is considered fundamental for quality teaching.

Designing and developing content in this time of crisis means shifting learning resources online. For a successful experience, the following are recommended:

- 1) Large scale collaborative efforts like Project Quantum^{48,49} (Oates, Coe, Peyton-Jones, et al., 2016b) where more than 8 thousand multiple-choice questions in the computing realm have been crafted and have undergone a rigorous quality check. These questions have been used with multiple pedagogical approaches such as peer-instruction allowing instructors to pose fundamental concepts questions aiming at fostering self and group reflections with live in-class activities.
- 2) Link competencies (Clear et al., 2020a) to learning resources (Maiorana, 2021d), (Maiorana, 2019b) and assessment activities (Giordano, Maiorana, Csizmadia, Marsden, Riedesel, Mishra, et al., 2015). This will allow for a tight focus of the educational intervention making it possible to develop the selected competencies.
- 3) Privilege interactive content. The goal can be pursued through many approaches up to changing the flow of learning resources design and development: start from formative assessment then proceed to discuss the assessment activities and deepen the material with an inquiry-based approach (Maiorana, 2021a), (Maiorana et al., 2021). The flow of activities in the online book will be formative assessment, discussion regarding the assessment, readings, project activities, summative assessment (Maiorana, 2021a).
- 4) Support the content with interactive books like the open-source Runestone project or commercial solutions such as zyBooks. Interactive books support the creation of many different assessment activities like Parsons problem where a solution to a given problem is broken into pieces and the learner must assemble the pieces in the correct order. Online books provide tools for designing and developing animations too; these animations can provide support for student self-reflection if enough context information is provided. Other open source interactive book platforms like Jupyter Book (Barba et al., 2019) have been successfully used in many disciplines and can be integrated with cloud-based proprietary platforms like Codio⁵⁰, a “flexible, accessible, and scalable platform for the Computing educational community”.
- 5) Provide a variety of activities (Maiorana, 2019b) in a variety of learning path (Maiorana, 2021d). In computing, examples range from designing a solution to coding it either with block-based or textual based languages, to using puzzle-based and unplugged activities. Organize these activities with a low floor entry point suitable for all students and a high ceiling supporting curiosity and challenging all learners. Allow for multiple learning paths along the content letting educators guide students to choose the learning path that best fits their context and zone of proximal development. Indicating these learning paths; offering an

⁴⁸ <https://community.computingatschool.org.uk/resources/4382/single>

⁴⁹ https://www.computingatschool.org.uk/custom_pages/107-quantum

⁵⁰ <https://sigcse2020.sigcse.org/attendees/supporter-sessions.html#codiobuildingscalablesolutionstoaddresssthechallengesofthecomunity>

annotated guide for a set of learning resources developed around a curriculum; and stating the level of difficulties of each activity will facilitate the students' self-reflection.

- 6) Offer a rich variety of delivering media: from short videos to interactive activities.
- 7) Design all the content in an accessible way providing support ranging from interactive video captioning to images' description (Cristaldi et al., 2020).
- 8) Leverage on Communities of Practices (Maiorana et al., 2022a), (Maiorana, Altieri, et al., 2020) like Scientix⁵¹, Computer Science Teachers Association (CSTA)⁵², Computing At School⁵³ where the members with different experiences support each other, offering a cascade of learning resources^{54,55}, scholarship reading and networking possibilities.

Online education (Maiorana et al., 2022a) can foster collaborations among learners and educators in an international and interdisciplinary setting (M. Caspersen et al., 2022b), (M. Caspersen et al., 2022a), (M. E. Caspersen et al., 2018a) and hence could represent a great opportunity for making Coding, Computational, Algorithmic, Design Creative and Critical Thinking, "Informational Thinking" and "Communicational Thinking" (Maiorana, Gras-Velazquez, et al., 2023) and technological education available for everyone by introducing Computing and Information Technologies in the context of other disciplines. This will be the topic of a special issue submitter to the IEEE Transaction on education available in the appendix of the thesis. Lots of real word applications and learning resources exist in this regard⁵⁶.

The rest of the chapter is organized as follow:

- 1) In section 4.1 we summarize published and under review works on comparing decades of computing enacted curriculum in three countries: Italy, United Kingdom and USA in
 - a. In section 4.1.1 Database and Project management(Maiorana, Csizmadia, & Richards, 2020a)
 - b. In section 4.1.2 Algorithms, Programming, Data and Computational Thinking; Network Internet and Security; Ethics (Maiorana, Csizmadia, & Richards, 2020b)
 - c. In section 4.1.3 Human-Computer Interaction

⁵¹ <http://www.scientix.eu/>

⁵² <https://www.csteachers.org/>

⁵³ <https://www.computingatschool.org.uk/>

⁵⁴ <http://www.scientix.eu/resources>

⁵⁵ <https://community.computingatschool.org.uk/resources/3084/single>

⁵⁶ <https://teachinglondoncomputing.org/interdisciplinary-computational-thinking/>

The importance of the topic areas selection finds research support in the recently released Informatics reference framework for schools (Caspersen, 2022).

- 2) In section 4.2 we present a curriculum and a learning path for a first course on computing addressing content pedagogies and technologies (Maiorana, 2019b), (Maiorana, 2021d). The curriculum design has been sharpened by its implementation in a lower secondary (Giordano, 2014) setting and in an undergraduate course for student majoring in humanistic studies (Maiorana, 2018). This is intended as a proof of concepts of the finding learned during the competencies design process presented in chapter 3 and in (Clear, 2020): the ability of competencies-based design to tailor curricula to the context of the degree programs.
- 3) In section 4.3 we discuss on learning path for computing and civic ed education as emerged from a panel and a paper under review along with consideration on the importance of social aspects in the educational practice. This fulfills educational need related to responsibility and empowerment, creativity and active digital citizenship become mandatory for Italian high school students and suggested in recent released Informatic Reference Framework for schools (Caspersen, 2022).
- 4) In section 4.4. we cover the design of a second course on algorithms and data structures covering aspect related to the design of the content (Maiorana, 2021a) pedagogical and technological approaches (Maiorana et al., 2021). The work leveraged of research experience with secondary students as a second informatics course and in an undergraduate setting inside the Computational Core experience at Kansas State University.
- 5) Finally, in section 4.5 we present a curriculum suited for a summer course at the Johns Hopkins University academic Center for Talented Youth on Foundations of Programming. The proposed leaning path can be viewed a glue of the above presented design experience.

4.1 International comparison of enacted computing curriculum

4.1.1 Database and Project management

The work (Maiorana, et al., 2020a) reports a learning trajectory (LT) aiming at developing competencies in database development, and project management. To do so, we leveraged on mandatory curricula in England, national guidelines in Italy, frameworks from the ACM and Computer Science Teacher Association. This curriculum rooted in design methodology has the flexibility to adapt to fast content changes, and the indication of minimum, advanced and elective competencies allows serving students with different needs with an inclusive and accessible approach. Interdisciplinary project experiences and various progression paths

give flexibility to LTs allowing their use for teachers' learning both pre and in service and for undergraduate studies.

From an Italian perspective, leveraging on a multi-year experience started in 2000 in teaching databases in a technical Liceo (Italian equivalent of high school) in Informatics (INF) , on pre-service and in-service teacher preparation course as well as in undergraduate and graduate courses we propose a learning path that integrates Database (DB), Web development (WD), Human-Computer Interactions (HCI), Cybersecurity (CS), and Project Management (PM) suggesting learning trajectory, pedagogies, and technologies suited for the different learning paths. For high school, the context refers to a 13th grade Italian technical Liceo majoring in CS (in Italy the school system requires five years of primary school, three years of Middle school and five years of high school) where learners attend CS for six hours/week, Networking and cybersecurity for five hours/week, System design (SD) four hours/week, and Project Management (PM) three hours/week.

The teachers' course follows the same schedule with a different pace and an emphasis on pedagogies and technologies and peer project revision. The undergraduate and graduate courses can be differentiated on the deepening level: core concepts and ideas (C) are considered mandatory for all the courses; elective (E) concepts for high school becomes progressively mandatory for undergraduate and graduate courses.

The innovative aspects of the CS curriculum are:

- 1) A simultaneous introduction of all core database concepts: entity relation design, logical model, SQL language, interface design from the first units; these concepts are refined and deepened presenting a new aspect for each unit while developing a course project.
- 2) A relatively large scale project developed, using an incremental approach, inside the curriculum.
- 3) An interdisciplinary approach (Cassel et al., 2014) focusing on using and comparing design principles and techniques drawn from several disciplines: databases, web programming, human-computer interaction, programming, security.
- 4) Use of a Project Management approach in all phases: from software project to security (R. Trilling, 2018).

The Project-based (Blumenfeld et al., 1991) and learning by doing (DuFour & DuFour, 2013) approach of the curriculum has been deployed across several years in many different projects: community service using App Inventor, NoSQL databases (Gray et al., 2012a), project management for social good (PMIEF, n.d.), query optimization with Microsoft Query Analyzer (Giordano & Maiorana, 2013g) and index selection (Maiorana & Giordano, 2013a)

professional certification path (Rowland et al., 2018), (Thernstrom, 2009)^{57,58,59}, real-life free open source project (H. J. Ellis et al., 2015b)⁶⁰, mobile web development (Maiorana, 2015), SQL puzzle-based learning (Viescas & Hernandez, 2014), (Celko, 2006). (Celko, 2010), security issues related to databases (Atzeni et al., 1999), (Elmasri & Navathe, 2011), (Taylor & Sakharkar, 2019) and database programming using Visual Basic for Application (VBA) as a scripting language.

Those projects engage students in group works adopting inclusive pedagogies to minimize inter and intragroup performance differences. For undergraduate and graduate courses, the most successful approach was student lead activities with real customers where the student engages in a real-life project in a self-chosen, small-medium enterprise in the local community. Many projects had continued in stages after graduation.

These adopted pedagogies aim at nurturing an active student role with an inquiry-based approach (Hazelkorn, 2015b), (Education et al., 2007), facilitating peer instruction (L. Porter et al., 2016b)⁶¹ or Process Oriented Guided Inquiry Learning (POGIL), ('Education Ambivalence', 2010)⁶². A flipped classroom approach (J. L. Bishop & Verleger, 2013) facilitates the participation of motivated students in classroom activities.

The works (Giordano & Maiorana, 2013g), (Maiorana & Giordano, 2013a), (Maiorana, 2015) and (Giordano & Maiorana, 2013d) report evidences supporting the curriculum design from stakeholders arose from survey and teachers' evaluations.

From an English perspective, the creation, maintenance, and manipulation of a database runs like a golden thread throughout the computing Programme of Study (DFE, 2013b) as learners initially access, interrogate and then create a flat file database. As they progress through their formal studies of computing, learners interrogate both flat file and relational databases using a Structured Query Language (SQL).

Finally, in their formal study of computing learners design, develop and deploy a relational database, using Codd's relational model as a framework (Codd, 1990), with a customised interface either as a standalone application or a backend system to support either a web based or a mobile based application. This is a practical manifestation of Bruner's "spiral curriculum" (Bruner, 1996a) applied to database development in which learners develop their mastery of database development. Thus, learners' confidence, capability and competence in interrogating, manipulating, design, developing and deploying both flat file and relational databases.

⁵⁷ Microsoft. *SQL Server 2017 Express Edition*. Available <https://www.microsoft.com/it-it/sql-server/sql-server-editions-express>

⁵⁸ Microsoft. *SQL Management Studio* Available <https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-2017>

⁵⁹ Microsoft. *Microsoft sample databases* Available. <https://docs.microsoft.com/en-us/sql/samples/sql-samples-where-are?view=sql-server-2017>

⁶⁰ *MusicBrainz Database Schema*. Available https://musicbrainz.org/doc/MusicBrainz_Database/Schema

⁶¹ *Peer Instruction for Computer Science* Available www.peerinstruction4cs.org/

⁶² *Process Oriented Guided Inquiry Learning (POGIL)* Available <http://guidedinquiry.org/>

The authors amended the programming learning taxonomy developed by Fuller et al. (Fuller et al., 2007a) to generate a learning taxonomy which is specific to database development as summarised in Fig. 1 and specific to project management in Fig. 2 in the references paper (Maiorana, et al., 2020a) .

Throughout the English computing curriculum, learners are introduced to principles of project management as they design, develop, and deploy computer systems, including database systems. As learners progress to study accredited computing qualifications, they develop their mastery of managing computing projects through applying and manipulating project management approaches, tools and techniques to the projects they manage. These approaches, tools and techniques include design sprints, dedicated project management software and agile methodology.

Table 1 summarize the content, pedagogies and technologies (Maiorana et al., 2017c), (Maiorana, Richards, Lucarelli, Berry, et al., 2019b), (De Rossi & Angeli, 2018a), (De Rossi & Trevisan, 2018b) of three strands: Computer Science (CS) focusing on database design, development and programming.

Whilst System Design (SD) focusing on Human Computer Interaction, Interface Design, and Web Programming; Project Management (PM) focusing on project management techniques applied to software and system development with an emphasis on workflow design (Sharp & McDermott, 2009) and project management (B. Trilling, 2014). A mapping to competencies proposed by the CSTA framework and standard (Seehorn et al., 2011), (CSTA, 2017d), (CSTA, 2016), (CSTA, 2020b), the English computing curriculum (DFE, 2013b), the Italian curriculum (Bellettini et al., 2014) and the ACM/IEEE Information Technology Curricula (ACM/IEEE, 2017) is proposed.

Table 1: The interdisciplinary learning trajectory.

U	Content (CS)	Content (SD)	Content (PM)	Tech.	Competencies.
1	E/R, entity, primary key, attribute (C)	User Interface for Parametric Queries (C)	Workflow design: swimline (C)	Access (CS, SD) Visio/DIA (PM)	(1,2,4,5),25; 29,30;10,13,15,23,37
	Projection and filtering using one table. (C) Regular expressions (M)	Forms (C)	Workflow design: flow diagram (C)	HTML VirtualBox	16,17,23,26,28;20,30;10,13,15,23,37;9;7
	User Defined Function (M)	Client-side data validation (C)	Workflow design: Petri Nets	Javascript (SD) Visio/DIA (PM)	1,18,22;13,15,23,37;9
	File programming (A)			VBA (CS)	18,22
2	Associations 1-N, 1-1; association attribute, multiple associations, association with role. (C). EER diagram	Usability principles, user interface design principle, usability evaluation (C)	Workflow implementation	Access (CS, SD) SharePoint Portal Server. Share Point Designer (PM)	1,2,4,5,25;29,30;10,12,13,15,23,37
	Join and set operations (M)	Server-side scripting: imperative programming	Project definition	PHP (SD) Visio (PM)	16,17,23,26,28;36,37
	Index, query plan and query optimization (A)			SQL Server Express	16,17,18,19;23,24
	Cursor, concurrency and locks (A)		Project Charter	VBA (CS) PHP (SD)	36,37
3	Associations N-N.	Use case (C). Usability evaluation (A)	OBS	UML (SD) Sharp&McDermott (SD/PM)	16,17,18,23,25;26,28;29,30;36,50
	Sub-queries. Nested and correlated sub-queries	PHP classes	WBS	PHP (SD, PM)	12,13,14;6,7,8
	Stored procedure to insert, update and delete information	Stored procedure use in PHP	Testing	MYSQL	18; 34,35
4	Ternary and n-ary associations	Cookies	Unit Testing	PHP (SD, PM)	1,2,4,5,25
	Group by and nested queries	Sessions	Pert	PHPUnit (PM)	
	Transaction	Transaction use in PHP	Resource plan	Microsoft Visio (PM)	31,32,33,34
	Use case and user interface design for ternary relationships		Gantt	Microsoft Project, Libre Project (PM)	29,30;31,32,33,34,35;36,37
5	Associative entity, Normalization. ELH.	Use case and user interface design for associative entity	Risk Management	Visio	25;29,30;36,37

	Having clause (C)	Websserver	Budget and financial plan	PHP	16,17,23,26,28; 28;32,33,34
	Stored procedure for user constraints & business rules			MYSQL SQLServer Express	1, 21,22,27

U	Content (CS)	Content (SD)	Content (PM)	Tech.	Compet.
6	Security: users, objects and permissions	Security management in PHP	Project development	MYSQL SQLServer Express	27; 36,37
	Recursive associations & queries	Cryptography	Project release	PHP	20,21
	Common expression table		Project revision	Access Data project	20,21;36,37
	Data security Language			SQL Server Express	27
	Triggers and their use				1, 21,22,27
	Index data structures	B-Tree in PHP		Query analyzer	19,21

Legend:

Italian Competencies for CS: 1) use strategies to afford problems and elaborate adequate solutions; 2) develop CS network applications and distance services; 3) choose device and tools on the basis of their functional characteristics; 4) manage projects according to procedures and standards 5) write technical deliverables and document the individual and group activities related to professional situations

Italian Competencies for SD: 6) develop CS network applications and distance services; 7) choose device and tools on the basis of their functional characteristics; 8) manage projects according to procedures and standards; 9) Configure, install and manage data and network systems; 10) write technical deliverables and document the individual and group activities related to professional situations

Italian Competencies for PM: 11) Identify and apply project management methodologies; 12) manage projects according to procedures and standards ; 13) use principles related to economy and management of process and services; 14) analyze value, limit, and risk of different technical solutions in relation with social and cultural life with a particular emphasis to security on professional work and safeguarding of the person and environment: 15) use and develop

CSTA K-12 Computer Science Standards Revised 2017 – Data and Analysis: 16) (3B-DA-05) Use data analysis tools and techniques to identify patterns in data representing complex systems; 17) (3B-DA-06) Select data collection tools and techniques to generate data sets that support a claim or communicate information

CSTA K-12 Computer Science Standards Revised 2017 – Algorithms and Programming: 20) (3P-AP-11) Evaluate algorithms in terms of their efficiency, correctness and clarity; 21) (3P-AP-12) Compare and contrast fundamental data structures and their use; 22) (3B-AP-13) Illustrate the flow of execution of recursive algorithms; 23) (3P-AP-15) Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution; 24) (3B-AP-18) Explain security issues that might lead to compromised computer programs.

National curriculum in England – Computing program of study: key stage 4. 25) develop and apply their analytical, problem solving, design and computational thinking skills; 26) understand how changes in technology affect safety, including new ways to protect their online privacy and identity, and how to identify and report a range of concerns

Information Technology Curricula 2017 - ITE-IMA Information Management (ITE-IMA domain competencies) 27) (B) Design and implement a physical model based on appropriate organization rules for a given scenario including the impact of normalization and indexes; 28) (C) Create working SQL statements for simple and intermediate queries to create and modify data and database objects to store, manipulate and analyze enterprise data; 29) (E) Perform major database administration tasks such as create and manage database users, roles and privileges, backup, and restore database objects to ensure organizational efficiency, continuity, and information security

Information Technology Curricula 2017 - ITE-PFT Platform Technologies (ITE-PFT domain competencies) 33) (D) Produce a block diagram, including interconnections, of the main parts of a computer, and illustrate methods used on a computer for storing and retrieving data.

Information Technology Curricula 2017 - ITE-UXD User Experience Design (ITE-UXS domain competencies) 34) (A) Design an interactive application, applying a user-centered design cycle and related tools and techniques (e.g., prototyping), aiming at usability and relevant user experience within a corporate environment; 35) (B) For a case of user-centered design, analyze and evaluate the context of use, stakeholder needs, state-of-the-art interaction opportunities, and envisioned solutions, considering user attitude and applying relevant tools and techniques (e.g., heuristic evaluation), aiming at universal access and inclusiveness, and showing a responsive design attitude, considering assistive technologies and culture-sensitive design

Information Technology Curricula 2017 - ITE-WMS Web and Mobile Systems (ITE-WMS domain competencies) 44) (A) Design a responsive web application utilizing a web framework and presentation technologies in support of a diverse online community; 45) (B) Develop a mobile app that is usable, efficient, and secure on more than one device; 46) (C) Analyze a web or mobile system and correct security vulnerabilities; 47) (D) Implement storage, transfer, and retrieval of digital media in a web application with appropriate file, database, or streaming formats; 48) € Describe the major components of a web system and how they function together, including the webserver, database, analytics, and front end

Information Technology Curricula 2017 - Global Professional Practice (ITE-GPP-08 Project Management Principles competencies) 49) (D) Evaluate related issues facing an IT project and develop a project plan using a cost/benefits analysis including risk considerations in creating an effective project plan from its start to its completion.

Information Technology Curricula 2017 – Software Development and Management (ITE-SDM Domain competencies) 50) (B) Use project management tools and metrics to plan, monitor, track progress, and handle risks that affect decisions in a computing system development process involving a diverse team of talents and professional experience

4.1.1.1 Lesson learnt

The main lessons learned from comparing and contrasting these international experiences in Italy, England and the United States of America are:

- 1) Focusing on design principles like data modeling both static and dynamic, workflow modelling through swimlane diagrams represent, in the long run, a definitive plus. The design phase, as reported in Table 1, is addressed from the first module.
 - 2) Use of easy to use tools that automate the majority of the process still allowing ample space for customization represents a way to reduce the burden of cognitive overload of a more abstract study related to design principles. As reported in Table 1 tolls Microsoft SQL Management Studio, available in the free express edition, simplify reverse engineering real case database design like the Adventure work. Share Point Portal Server and Share point Designer is suggested for workflow prototyping.
- Coupling design tools with developing tools like Microsoft SQLServer Management Studio Express Edition and its Query Analyzer allows learners to better understand algorithms that underpin major SQL operations. For example, converting a query to the underlying algorithm in a procedural programming language representing a relation between Computer Science (CS) and Information Technology (IT) courses. Use of these tools and the interplay with the design.
 - Use of a class site supporting content sharing, class discussion, interactive activities, and self-reflection beyond class time is a useful pedagogical tool for developing both communication and collaboration between students.
 - Students are offered low floor activities that, with a strong foundation on rigorous concepts and techniques, allows students to easily grasp the core concepts and apply them to practical projects. In addition, high ceiling activities are suited to challenge

students who progress at a faster pace than others (Nussloch et al., 2014) preventing boredom. Thus, allowing for quality and inclusive education for all.

- By providing multiple learning paths and progression pathways (Dorling & Walker, 2014), (Maiorana, 2021d) coupled with the use of different types of activities, ranging from unplugged (Bell et al., 2012b) to design and coding, it is possible to engage students with different learning styles with the best match of educational activities and students' zone of proximal development. For example, as reported in table 1, for module 1, design activities either with pencil and paper or with technological tools, are coupled with the design implementation and with coding using VBA.
- Project management techniques provide all students with a set of competencies and skills that can be used across all disciplines, starting from IT and CS, and in everyday life.

- Dual enrolment program with academia and professional certification path, such as the Microsoft certification path related to the tools suggested in Table1, represent a good way to mediate between work request of students, especially in a low-income environment, and student progression to academia, always clearly stating and advocating for a lifelong learning process, ideas clearly embraced by industry and academia.

4.1.2 Algorithms, Data, Networks, Security and Ethics

The paper (Maiorana, Csizmadia, & Richards, 2020b) examines three primary competencies, which are: 1) Algorithms, Programming, Data, and Computational Thinking (CT); 2) Networks, Internet and Security; and 3) Ethics. Due to the standards and relationship of competences, the authors categorized, algorithms, programming, data and CT together. Thus, we leveraged national guidelines in Italy, mandatory national computing curriculum in England, and state guidelines in Alabama, USA in conjunction with ACM computing frameworks.

4.1.2.1 Programs, Algorithms, Data and CT

Italy. Leveraging upon a multi-year experience which started in 2000 (Giordano & Maiorana, 2015b), (Giordano & Maiorana, 2014c), (Maiorana, 2019b), (Maiorana, 2021d), it was proposed that a learning path be used. This learning path would include puzzle-based, unplugged, and computer-based activities. The activities involved a variety of tools and technologies, ranging from block-based to text-based languages, covering digital literacy, algorithms, and data structures. This fosters competencies (Nardelli, Forlizzi, Lodi, Lonati, Mirolo, Monga, Montessor, et al., 2017) summarized in Table 2, assessed and supported by software artifacts developed by classmates in a shared repository of solutions. For high school, the context refers to a 9th – 10th grade Italian technical Liceo majoring in CS and in a Science liceo focusing in Mathematics and Science. In Italy the school system requires five years of primary school, three years of low high school and five years of upper high school. In their first biennium students attend three hours of computing a week in the technical liceo and two hours a week in the Mathematics and Science liceo. The suggested learning trajectory is to engage students with unplugged activities and puzzle based challenges. Then students will transform the challenge into coding solutions offered by the teachers will generate a coding solution for the specific challenge. Documentation of the coding solutions could include natural language, flow diagram, pseudo code, block-languages and, at the end, text languages. Engaging activities include reading and comprehension of code, commenting code, translating from one paradigm to another, e.g. block-based to text-based language, flowcharting, debugging, fill in the gap, modifying an algorithm ensuring that students can successfully design and code a functional solution. This learning trajectory focuses upon core concepts of abstraction, generalization, and functional decomposition. Design practice at the

beginning of the course gives teachers time to guide and support students to become independent learners and take ownership of these concepts. From observation, this avoids cognitive overload at the end of the course when tiredness and deadlines escalation prevent a great learning experience. For those schools majoring in Computing, the second biennium and fifth year of high school experiences range from object-oriented to database design. Schools that do not offer Computing as a standalone subject offers access to computing as a breadth across the curriculum.

England. The Computing Programme of Study's opening sentence places CT along with creativity at its kernel (DFE, 2013b). In the context of this framework, CT's components are identified as: algorithmic thinking, decomposition, generalisation (patterns), abstraction and evaluation (Csizmadia, Curzon, Dorling, et al., 2015) and further amplification is provided in *Progression Pathways* (Dorling & Walker, 2014). CT concepts, practices and perspectives (Brennan & Resnick, 2012b) are introduced, developed and refined by engaging in both unplugged and plugged activities, such as Bebras Challenge, physical computing, and coding activities. Within high schools at Key Stage 3, all students are expected to develop mastery of both a block-based programming language and a textbased programming language. This is achieved by being able to create and program a solution to a given problem, utilizing either standard algorithms or algorithms they have created themselves, and manipulate appropriate data structures. At Key Stage 4, students self-select to study computing qualifications awarded by external Awarding Organisations. These students continue to develop their proficiency in applying CT to increasingly more complex programming problems, selecting appropriate algorithms and data structures to create a unique digital solution to each problem they encounter. Additionally, all computing students are expected to document the design, development and deployment stages of their solution using appropriate methodologies, such as PRIMM (Sentance et al., 2019b).

Alabama, USA. To provide stronger student outcomes, direct instruction is transitioning to inquiry-based and discovery learning strategies. ADLCS (ADLCS, 2018) incorporates five Content Standard Strands with Topics. Abstraction, algorithms, programming, and development are CT topics. Students progress from concept to CS skills that will achieve deeper understanding in programming, algorithms, data structures, and CT (ADLCS, 2018), (NIST, 2014). Because ADLCS is integrated into the existing curriculum, P-2 students learn digital literacy and basic computing skills such as keyboarding. Grades 3-5 begin to slowly transition students into computing concepts with grades 6-8 becoming emerged in learning algorithms, coding, and databasing. However, ADLCS provides schools with great latitude to develop courses and strategies for continuing computing education in grades 9-12. Although ADLCS (ADLCS, 2018) is designed for equity in access, issues like connectivity have tasked educators to find appropriate methods to teach these competencies with and without access to a computer or the Internet. Coding competency is developed using both Scratch and Texas Instruments Innovator Rover for instruction. Concepts in engineering design creation, and

other STEM+C projects are expected to be utilized in age appropriate classroom instruction. Teachers use their professional judgement to determine which technologies are most appropriate for their students. Table 2 outlines the levels of competencies, knowledge, and skills.

4.1.2.2 Networking, Internet and Security

Italy. The proposal (Nardelli, Forlizzi, Lodi, Lonati, Mirolo, Monga, Montessor, et al., 2017) encompasses safeguarding individual data and protection company data including intellectual property. Students should evaluate the reliability of the content and have experiences with retrieving and organizing data and metadata in simple contexts like HTML and data description languages. Experiences in such directions range from scaffolding activities on how to perform a bibliographic study (Maiorana, 2019b) to national competitions on web searches⁴. In the last three years of high school, students majoring in computing, networking and security explore at a deeper level mathematics, specialized areas of computing, client-server programming, cybersecurity issues, network protocols, and web services design and implementation.

England. The topic of data communications and networking is a strand which runs throughout the computing curriculum as a golden thread (DFE, 2013b) and is regarded by Dorling & Walker (Dorling & Walker, 2014) as one of the five themes of the computing Progression Pathways framework. K-12 teachers identified this aspect of computing as a challenge to teach (Sentance & Csizmadia, 2017b) and the topic is referred to as the Cinderella of the computing curriculum (Csizmadia & Maidens, 2018). Therefore, a number of approaches have been developed ranging from unplugged activities, such as Network Protocols (Bell et al., 2012b) to physical computing activities, such as Binary Boxes (Csizmadia & Maidens, 2018). Subject matter experts, including one of the authors, developed a networking curriculum sponsored by Cisco Systems and hosted on the Cisco Network Academy eLearning platform, which utilizes Packet Tracer (Janitor et al., 2010) as a simulation tool for data communications and networking fundamentals. The Computing Programme of Study (DFE, 2013b) doesn't explicitly refer to cybersecurity, it is referred to within the formal specifications for national computer science qualifications at Key Stages 4 and 5. Exploratory learning activities that engage students with cybersecurity concepts and principles include physical computing, such as Man in the Middle Attack using Binary Boxes (Csizmadia & Maidens, 2018), escape room scenarios, and national programs. The National Cyber Security Centre's CyberFirst program organizes age-appropriate cybersecurity clubs for students and a girls-only cybersecurity competition, and out of class learning, such as Government Communications Headquarters' (GCHQ) Summer Schools for Key Stage 4 students. In addition, teaching cybersecurity within the computing curriculum provides teachers with an ideal opportunity to discuss moral, social,

legal and ethical issues associated with the student's own digital identity and their role and responsibilities as digital citizens.

Alabama, USA. Online safety and security of personal information are discussed throughout P12 courses. In 9-12 students learn more about encryption, steganography, network design, and various cybersecurity principles (ADLCS, 2018). NIST and NICE recommended standards and curricula that could provide lessons and assessments bridging the gap between education and industry (NIST, 2014)⁶³, (Richards & Turner, 2019). Due to COVID19 and the closing of schools, (ADLCS, 2018) has not been fully implemented in a traditional classroom. However, P-12 schools have rolled out additional virtual learning platforms providing a more robust environment to engage students and incorporate the ADLCS standards. NIST (NIST, 2014) standards for these topics that could be easily adopted for the classroom and assessments for modeling the role of network protocols in transmitting data across the Internet. The inquiry-based lessons would employ the process for security and privacy control assessment to determine artifacts, milestones, stakeholder communication channels, etc. Once the artifacts are prepared for assessment, students undertake at least one recommended test to demonstrate their competency, understanding and skills. Then debriefings and plans would build on the outcomes achieved with a post-assessment to replicate real-world scenarios. The focus of cybersecurity initiatives are pursued in Italy, England, and Alabama through summer camps and other activities.

4.1.2.3 Ethics

Italy. The proposal (Nardelli, Forlizzi, Lodi, Lonati, Mirolo, Monga, Montessor, et al., 2017) emphasizes ethical digital competencies and reflections on the influence and consequences of computing technologies on society. Digital ethics are integrated across the whole curriculum to build tolerance and participation. International activities such as then Safer Internet Day or national initiatives against bullying⁵ in all its forms are undertaken.

England. Woven throughout the Computing Programme of Study is the theme of digital literacy (DFE, 2013b) which encompasses digital ethics and promotes digital wisdom. There are national initiatives, such as Anti-Bullying Week which promote how to behave sensibly and safely online. Child Exploitation and Online Protection (CEOP) programs raise awareness of how learners could be exploited online and how they can report such incidents. As part of digital literacy, students learn the impact of their digital footprint, copyright, plagiarism and the requirement to acknowledge and attribute information sources they have used in any work they have produced.

Alabama, USA. Like Italy and England, Alabama emphasizes legal and digital ethical principles through curriculum (NIST, 2014). Topics include protecting personally identifiable information (PII), types of cyberattacks, legal outcomes of landmark cases, intellectual

⁶³ National Standards. (2019). Retrieved May 24, 2019, from <https://www.educationworld.com/standards/>

property and clients' rights. In addition, students are taught digital citizenship and in some schools are required to sign a contract to actively practice positive digital citizenship.

4.1.2.4 Mapping to standards

We provide a comparison of the three primary areas discussed in this paper against the computing education standards in Italy, England, and Alabama, USA (see Table 2: Comparison of Computing Standards). The listed standards are the primary ones the authors feel address a baseline computing curriculum.

4.1.2.5 Comparison and Lessons Learned

Italy: In Italy and Europe (M. E. Caspersen et al., 2018a) two main directions are pursued. computing is introduced as a standalone subject discipline or is integrated in the context of other disciplines (Maiorana, 2019b), (Nistor et al., 2019). This second approach is necessary to enhance the level of computing education that exists in other countries. Unlike Italy, where the compulsory study of computing and its formal assessment is not enshrined in national law but relies on national and European initiatives (Barendsen et al., 2015), (Falkner, Sentance, Vivian, et al., 2019b), (Falkner, Sentance, Vivian, et al., 2019d), (Forlizzi et al., 2018b), (Giordano, Maiorana, Csizmadia, Marsden, Riedesel, Mishra, et al., 2015), (Kearney & Gras-Velázquez, 2015), (Licht et al., 2017b), (Nistor et al., 2019).

England. The Royal Society has published two state of the nation reports for computing education. The first, *Shut Down or Restart* (Britain, 2012), was the clarion call to establish computing as a discipline within the English national curriculum. While *After the Reboot* (R. Society, 2017), critiqued computing's implementation nationally, and made 12 recommendations to support computing's growth at local, regional and national levels. Additionally, *The Roehampton Annual Computing Education Report* (Kemp & Berry, 2019) analyses governmental statistics to identify national trends within computing education. While Computing At School's annual survey indicates teachers' perceptions of the subject's local implementation, including successes and challenges in teaching computing (Sentance & Csizmadia, 2017b). The survey results are considered during program reviews and qualification changes.

Alabama, USA. Although the federal government has been urging the adoption of *Computing for All* (Maiorana et al., 2017c), (Maiorana, Richards, Lucarelli, Berry, et al., 2019b), (NIST, 2014)⁹², (Nistor et al., 2019) (Code.org, 2018), states have not moved as rapidly, Alabama blended digital literacy and CS standards (Alabama State Department of Education, 2018b), rather than create a standalone program. Therefore, teachers have been charged with the integration of those standards into all curriculum for all grades (NIST, 2014), (Richards & Turner, 2019). Currently, Alabama is at the early stage of adoption of a computing curriculum and the development of degree programs in computing education for educators.

4.1.2.6 Lessons learned.

The main lessons learned from these international experiences in Italy, England and Alabama, USA are:

- 1) Focusing on applying design principles with the use of age and grade appropriate tools that automate the majority of the design process will reduce the burden of teaching design concepts.
- 2) Offer students low floor, high ceiling, and wide walls activities, with a strong foundation on rigorous computing concepts and techniques. Thus, allowing students to easily grasp the core concepts and apply them to practical projects. High ceiling activities are suited for students progressing at a faster pace (Maiorana, 2019b) while maintaining engagement and enthusiasm. The adoption of this approach allows high-quality and inclusive education for all.
- 3) Dual enrolment programs with academia and professional certification pathways permit students, especially from low socio-economic environments, to progress in computing education. These programs support learning and innovation promoted by industry and academia.
- 4) Diverse adoption of different computing education standards creates difficulties for institutions of educator preparation and professional development to provide teachers unified knowledge and skills in the computing curriculum.
- 5) Lack of compulsory computing curriculum within high schools create a deficit of competencies, knowledge and skills to start undergraduate studies.
- 6) The necessity to build interest among P12 students to become computing educators has become a challenge for institutions of educator preparation.
- 6) Unification of computing curricula which should be modelled on ACM/IEEE Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science (ACM/IEEE, 2013) with the NIST (NIST, 2014) recommendations to create a global level of competencies, knowledge, and skills.

Table 2: Comparison of competencies on Algorithms, Networking, security and ethics in Italy, England and Alabama, USA

Knowledge Categories	Required High School Competencies		
	Italy (Nardelli et al., 2017a)	England Standards (DFE, 2013a)	Alabama, USA DLCS (Alabama State Department of Education, 2018b)
Programs, Algorithms, Data, and CT	<p>Competences: T-S-1 - 10; T-S-16 Understand executor capabilities, ability to understand nature of problems and design general, correct, effective and efficient algorithms and associated data structures in a creative manner</p> <p>Knowledges Algorithms O-S-A-1 - 4 Fundamental algorithms; Algorithm comparison; Computability; Executor limit</p> <p>Programming O-S-P-1 to 7 Program structure; tracing; conditions; loops; variables; modularity; textual programming</p> <p>Data O-S-D-1 to 3 comparing data structures and data representations; data and metadata representation</p> <p>Digital creativity: O-S-R-1 Programming for expressiveness</p>	<p>Key Stage 3: Design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems</p> <p>Key Stage 3: Understand several key algorithms that reflect computational thinking [for example, ones for sorting and searching]; use logical reasoning to compare the utility of alternative algorithms for the same problem. Key Stage 3: Use 2 or more programming languages, at least one of which is textual, to solve a variety of computational problems; make appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular programs that use procedures or functions.</p> <p>Key Stage 4: Develop and apply their analytic, problem-solving, design, and computational thinking skills.</p>	<p>Algorithms/Programming: 1.2.1-1.2.2, 1.3.1 - 1.3.2, 2.4.1 - 2.4.39 Differentiate between a generalized expression of an algorithm in pseudocode and its concrete implementation in a programming language; demonstrate code reuse by creating programming solutions using libraries and API</p> <p>Data And Analysis: 1.1.4.1-1.1.4.2, 2.3.1-2.3.10 Compare/contrast basic data structures; develop models that reflect methods, procedures; summarize compression and encryption, use data analysis tools and techniques to identify patterns in data representing complex systems.</p>
Networking Internet & Security	<p>Competences: T-S-11 - 15 Recognize the nature of computer based systems and combine programming, services and systems to develop projects in accordance to user needs, ethical and societal concerns.</p> <p>Knowledges</p> <p>Digital Awareness: O-S-N-1-5 Sensors and devices for data collections; Protection of data privacy and security; Reliability of content; End-user requirements; Reuse and modification of programs and content</p> <p>Digital creativity: O-S-R-2 Combine programming and services</p>	<p>Key Stage 3: Design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems.</p> <p>Key Stage 3: Understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems.</p>	<p>Impacts of Computing: 1.1.1 - 1.1.2 Use an iterative design process, including learning from mistakes, to gain better understanding of the problem domain; problems not solved by humans or machines alone - discuss options to decompose tasks into sub-problems for humans or machines to accomplish. Computing Systems, Networks, and Internet: 1.5.1 - 1.5.2, 2.2.1 - 2.2.10 Evaluation of scalability and reliability of networks by relationship between routers, switches, servers, topology, packets, or addressing and the issues impacting network functionality; appraise the role of artificial intelligence in guiding software and physical systems-predictive modeling.</p>

Ethics	Competences T-S-13-14 Influence of computing systems on economics and society; Ethical and societal consequences of Information Technologies	Key Stage 3: Understand a range of ways to use technology safely, respectfully, responsibly and securely, including protecting their online identity and privacy; recognise inappropriate content, contact and conduct, and know how to report concerns. Key Stage 4: Understand how changes in technology affect safety, including new ways to protect their online privacy and identity, and how to report concerns.	Impacts of Computing: 1.1.2, 2.5.1 to 2.5.16 Explain how technology facilitates disruption of traditional institutions and services; explain necessity of Acceptable Use Policy, identify laws regarding use of technology and their consequences an implications including hacking, intellectual property, ransomware, and PII;
---------------	---	--	--

4.1.3 Computer Systems and Human Computer Interaction

The aims of the work (Maiorana, Csizmadia, et al., 2023) are to present ideas related to implementing computing education curricula for those who are about to embark on implementing a computing curriculum locally, regionally, and nationally. This work, currently under review, presents, as an extended case study, experiences run by the authors in England, Italy and USA, with a particular emphasis on the states of Alabama and Kansas, related to the teaching and learning of both Computer Systems and Human Computer Interactions, at the K-12 level. These experiences are related to teaching computing to school students, student teachers, pre-service and in-service teachers and industry led certification paths. The enacted K-12 computing curricula are framed by the respective computing curricula’s frameworks of each country. Specifically, mandatory national computing curriculum in England, national guidelines in Italy, and state guidelines in Alabama, USA derived from computing standards proposed by CSTA, ACM, and ISTE.

Rationale of exploring Computer Systems and Human Computer Interface

In this work we will compare the educational landscape in the three geographical areas in relation to Computer Systems and Human Computer Interface. We have chosen these two fields since they are less explored in literature, are less taught in high school curricula, and are strongly related, e.g., the design of user interface in operating systems. Recently the interrelation of HCI with a broad range of key aspects in computing such as adoption and use of information technology and user experience has been reviewed (Hornbæk & Hertzum, 2017). The importance of HCI in education, hence covering all aspects and domains related to education, including computing, has been highlighted by the adoption of universal design principles in education (S. E. Burgstahler & Cory, 2010b). A user centered approach, a foundational approach is user interface design, has been suggested as a problem solving approach advocating for a design thinking as a key competency for K-12 students of the 21st century.

Among the different domains of Computing Systems, we will focus on Computer Architectures (CA) and Operating Systems (OS).

Computer Architectures

England

- 1) From an English perspective, at Key Stage 3, learners study computing as part of a mandatory national curriculum. Learners are required to not only understand how hardware and software components are combined to create a functional computer system, but how those components communicate with one another and other systems. Learners may engage in a range of learning activities such as The Human Computer (unplugged kinesthetic) (Bell et al., 2009), quizzes (peer instruction) (Csizmadia et al., 2019b) (Zingaro & Porter, 2014a), (Maiorana et al., 2015b), peer mentoring (Sentance & Csizmadia, 2017c) and constructing computing systems using either decommissioned computing equipment or the single-board computers such as Raspberry Pi, or BBC Micro:bit (physical computing) (Hodges et al., 2020), (Kalelioglu & Sentance, 2020), (Upton et al., 2016). At Key Stage 4, learners can choose to study different aspects of computing, such as computer science, creative media, as a formal qualification. The qualification they have chosen to formally study will determine at the level they will study computer architecture, from a black box approach to an in-depth examination of the components of a system and how they communicate with each other.

Italy

From an Italian intended, i.e. stated in national guidelines, perspective. We must differentiate between compulsory education up to K10 and the last three years of education from the 11th grade to the 13th grade. In compulsory education, students must understand the hardware and its capability as an executor to obtain a general view of the entire computer system from hardware to software. The national guidelines and proposal can be framed inside a learning path where students acquire competencies, knowledge, and skills. Students start from “recognizing the presence of computers in technological devices of everyday life”, proceed with “knowing the main hardware and software components of those devices” at the end of fifth grade. In lower secondary schools they proceed with knowing the architectural principles of computer systems in relation to external devices as well. By the end of the first biennium of high school, students proceed in broadening their view, considering the relation between the hardware, considered as an executor, and the algorithms they design which run using the underlying hardware of the computing system.

Students engaged in the computing major during the last three years of high school study computer systems in much more detail, covering deeper aspects of the hardware, its

programming in assembly language is covered in the 11th grade, the interface with sensors, actuators and microcontroller in relation to programming aspects is deepened in the 12th and 13th grades. The interface between the underlying hardware and the operating system is covered at the 12th stage, while HCI and user design principles for Computing Systems are covered at the 13th stage.

From the authors' experience, i.e. the enacted educators' perspective. In order to teach computer architecture, a scaffold learning pathway has been developed that proceeds according to stages in increasing levels of difficulties (Maiorana, 2021e), (Maiorana, 2019c). The first stage leverages on experiences gained from mobile computer science principle projects (Rosato et al., 2017b) where a set of supporting apps, such as a low-level simulator, were constructed. Learners watch an animation of the fetch, decode, execute cycle and after watching the animation have to design a set of simple instructions expressed in natural language (e.g. read data from a location, store data into a location, perform an operation) to describe the stages they observed in the animation. The animation increases in level of difficulty and spans different abstraction levels allowing both the static model and its dynamic execution to be explored. This approach adopts flipped learning as its content methodology (Maiorana, 2021b): learners start from reflecting on a topic and are guided through a set of formative assessment activities in which they describe in a formal way what they have seen in the animations. The approach has been developed inside a complete enacted curriculum (Maiorana, 2021e), (Maiorana, 2019c) suitable for the first two years of upper secondary schools. It represents a way to develop the competencies related to computer architecture and hardware of the Italian proposal for a National Computing Curriculum (Nardelli, Forlizzi, Lodi, Lonati, Mirolo, Monga, Montesor, et al., 2017b) and an European computing framework (M. E. Caspersen et al., 2022), such as understanding the capabilities of an automatic executor to express algorithms in an unambiguous way. Proposed elective paths can further explore parallel architecture (Goncharow et al., 2019), (S. Prasad et al., 2015), (S. K. Prasad et al., 2018). The approach can be used as a starting point for a computing curriculum for learners attending the last three years of upper secondary schools where they must acquire competencies in designing algorithms and implement them using a low-level abstraction programming language, such as assembly code (Abel, 2000), and interface a computer with external devices such as sensors and actuators using serial and parallel interfaces.

Alabama, USA

Students begin to learn and use digital devices with different operating systems (Alabama State Department of Education, 2018a) in Alabama in kindergarten. By high school, students are expected to have knowledge in network protocols, hardware architecture, and cybersecurity principles with an introduction to HCI concepts. The curriculum in high school includes learning and building skills in scalability, modeling and simulation, operating systems, life cycle of systems and applications (Alabama State Department of Education,

2018a). Six months into the integration of the new DLCS standards into the curriculum, COVID-19 caused a disruption in courses, which created challenges. However, schools can employ simulation applications that are grade and age appropriate, such as CPUlator®, EDUCache®, and EasyCPU®, to teach students in Computer Systems and HCI Perspective in Teaching Computing in England, Italy, and USA a traditional or virtual setting. For instance, CPUlator® is a web based simulator that contains instruction sets Nios II, ARMv7, and MIPS with the ability to test I/O devices, debug, debug assertions, and will accept input from assembly source code or ELF executables (CPUlator, n.d.). At the time of writing, fewer than 25 teachers in Alabama have been certified under the new Computer Science Education Teacher Certification.

Operating Systems

England

Most computing students at Key Stage 3 are taught operating systems from a theoretical perspective due to the reluctance of school IT technical staff to provide access to operating systems on either physical computers or virtual machines. However, in England at Key Stage 4 with formal computer science qualifications, learners will study both computer architecture and operating systems not only from a theoretical perspective but also from an experiential learning perspective, applying an active learning approach (McConnell, 1996) using visual simulators, such as Little Man Computer (Osborne & Yurcik, 2002), (Yehezkel et al., 2001), for learners to understand how both hardware and software components collaborate to create a functional theoretical computer based on the general von Neumann computer architecture model (Sentance & Waite, 2017). Innovative computing educators have recycled obsolete computing devices in order for learners to become familiar with building physical computer systems and then configuring the operating system to create a fully functional device. The computing recycling effort is pursued by public open call for donations from large state organizations.

Italy

From an Italian intended, i.e. stated in national guidelines, perspective. Students start their learning journey by using the operating systems managing the information technology systems, and proceed, by the end of primary school, to differentiating between the underlying hardware and the services provided by the operating system or the software infrastructure. By the end of lower secondary school, students understand the main functional concepts of computer based systems and devices. By the end of the first two years of high school, students are able to combine computer programs and software services offered, including the operating system ones, to develop well-structured informatics projects.

Students engaged in the computing major during the last three years of high school study operating systems in greater details in the 12th year, the second in the last triennium, covering

the history, the main types of operating systems, their classical structures from process and threads and their scheduling to memory and resource allocation, from file systems to device management, also covering aspects related to designing and developing programs which use system services.

From the authors' experience, i.e. the enacted educators' perspective. Operating Systems can be taught with the goal to develop digital competences in learners in the first biennium of upper secondary schools. In order to achieve this goal in (Maiorana, 2021e), (Maiorana, 2019c) it is suggested to seek students' engagement in searching for a solution instead of relying on a sequence of steps to perform a task. By leveraging sound principles of interface design, students are guided to explore and find ways to resolve the task at hand thus avoiding learning sequences of actions with a fast-changing rate according to the different software versions or, with the same software version, to the device used.

In the last triennium of the upper secondary schools, the focus shifted to teaching the design principles of modern operating systems. The suggested approach is to combine Computational Thinking mind tools, and Algorithm thinking strategies to enable and empower students to self-reflect and compare designing principles. Developing simulators such as queuing and scheduling systems, process and threads simulators has proven a successful approach allowing educators to nurture a combined set of competencies from algorithms, programming, and data and information areas and, in general, all the areas highlighted in the Italian computing curriculum proposal.

Alabama, USA

As students progress through high school, their knowledge of operating systems is expanded from theoretical concepts to the differences between operating systems, algorithms, programming, and how networks interact with systems. As early as 2005, discussions took place on teaching how to troubleshoot or tune a kernel. These technical areas can be extremely confusing unless students have the ability to access and configure the operating system to practice their skills (Nieh & Vaill, 2005). Today, one NSF funded virtual simulator moves the student from concept to skill (Buck & Perugini, 2019). However, the simulator must be downloaded and installed in order for students to engage with it. As simulation applications become more sophisticated, developers will need to provide documentation and user-friendly exercises to assist P12 educators with limited training in computing education topics. Currently, one university in Alabama is in the process of creating a system to assist their P12 partners in meeting the new requirements.

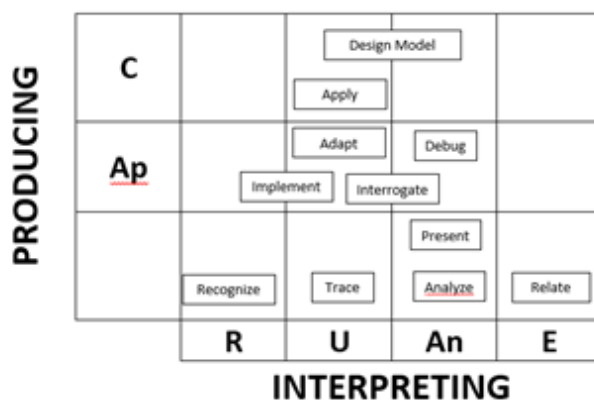
Human Computer Interaction

England

From an English perspective, at Key Stage 3, learners use, modify and create both text-based and graphical human computer interfaces (Franklin et al., 2020), (F. Martin et al.,

2020), (Sentance & Waite, 2017), (Sentance et al., 2019a) as they design, develop, and deploy digital artifacts as solutions to specific problems they are asked to solve. These digital artifacts include scripting a functional website using HTML, and possibly both CSS and JavaScript, programming both text-based interfaces and graphical interfaces, designing, developing, and deploying a mobile app using tools such as App Inventor (Gray et al., 2012b), (E. W. Patton et al., 2019). As learners progress through their formal studies of computing, they are introduced to principles of universal design as they design increasingly more complex interfaces which comply with accessibility guidelines.

In an optional out-of-class learning challenge, learners gain practical experience of designing a human computer interface to meet the needs of a client. For example, within the AWS GetIT App Challenge, female learners collaborate to design a mobile app to address a local ethical issue identified by the learners themselves. While in Arm’s FXP Extended Challenge where learners gamify a solution to one of three global problems linked to the United Nations Sustainable Development Goals of Zero Hunger, Good Health and Wealth. Finally, at Key Stage 4, in their formal study of a computing qualification, learners will design, develop, and deploy a customized and accessible interface for an application to solve a problem prescribed by the qualification’s awarding organization. This is a practical manifestation of Bruner’s “spiral curriculum” (Bruner, 1996b) as learners develop their mastery of designing customizable and accessible interfaces. Thus, learners’ confidence, capability and competence in designing accessible interfaces are developed. The authors amended the programming learning taxonomy (Fuller et al., 2007b) from Fuller *et al.* thus creating a Human Computer Interaction learning taxonomy as indicated in the following Figure.



Key: **C**reate, **Ap**ply, **R**emember, **U**nderstand, **An**alyze, **E**valuate

Figure 1 – Human Computer Interaction Learning Taxonomy

Italy

From an Italian intended, i.e. stated in national guidelines, perspective. During the first three years of primary school, students “develop a positive attitude towards computer-based

applications” and are able to navigate the interface of those applications and manage the interaction with them. By the end of fifth grade students are able to choose “suitable environments”. The choice also takes into consideration aspects related to HCI. These aspects are used to “create simple computer applications for expressive purposes”. During the three years of lower secondary school students sharpen their skill in selecting the best tool to fit their expressive purpose creating stories, games and creative software applications. They are able to design presentations applying basic interface and usability design principles. By the end of the first biennium of higher secondary school, students understand the importance of user needs for the implementation of their applications.

Students engaged in the computing major during the last three years of high school study design principles of software system and web applications in the 13th year, the last in the triennium. They study the design principles of software applications and systems, web application interfaces, mobile applications interface and responsive design principles. Care is devoted to accessibility issues and universal design principles involving the students welcoming special ability students, e.g., blind or deaf students, and the teachers through the national teachers professionalization and certification path for teaching to special students with special educational needs.

From the authors’ experience, i.e. the enacted educators’ perspective. By leveraging Universal Design principles (S. E. Burgstahler & Cory, 2010b) and accessible design, HCI concepts have been taught using a project-based approach requiring students first to evaluate and then to design an interface with a special focus on accessibility issues. A possible learning pathway starts with evaluating a web site with particular emphasis on accessibility and proceeds by designing and developing accessible web pages using formatting languages such as HyperText Markup Language (HTML5) and Cascading Style Sheets (CSS). This project presents the major design principles (Dix et al., 2003), (Ross & Freeman, 2022) which are applied in designing the interface of each software artifact developed by students.

Alabama, USA

Gamification of courses has become popular with some educators and students. In elementary schools, students are introduced to HCI through interactive games, Google Expeditions, National Geographic Explore, and other educational programs. Interactive digital media and simulations will engage students. Instruction in HCI applications requires more than skills and concepts. Ethics, laws, cybersecurity protocols in the creation and use of HCI and Artificial Intelligence are to be interwoven into the courses (Churchill et al., 2013) and will integrate GenCyber First Principles and Concepts wherever possible.

Mapping to standards

This section provides a comparison of the two areas of Computer Architecture and Human Computer Interaction and their sub-areas as discussed in this paper against the computing education standards in England, Italy, and Alabama, USA (see Table 3: Comparison of Computing Standards). The standards listed in the table are the primary computing standards the authors feel address a baseline computing curriculum.

From the table we highlight the emphasis on abstraction and its use to obtain a unifying view of computing systems. Abstractions allow us to see the interconnections between hardware and software, how the underlying hardware affects the computation process from program design up to computability considerations. Use of animations as formative assessment tools in a flipped approach has shown encouraging results in reaching different abstraction levels.

Regarding the curricula, while computer system has a long-standing tradition HCI is relatively new but now is present in all standards with an emphasis on accessibility issues. The last few years in Italy have seen numerous special educational activities for pre and in-service teachers carried out at the national level and teachers dedicated to special education have been hired in all public schools. In the USA curricula such as the Exploring Computer Science addresses competencies and content related to HCI and web design.

Lesson learned

From the authors' own experiences of teaching computing, the centrality of teaching principles of human computer interaction and design methodologies for human centered interfaces has increased over the years. A common approach amongst the authors is using accessibility issues to introduce design principles. This can be done and has been done in many courses: from an initial course in computing to specialized and advanced courses such as database courses, Information systems management courses and web design courses. Fast prototyping tools can be used to introduce interface design principles:

- 1) app inventor was successfully used to presents and apply design principles in an initial course on programming in different contexts, from high school courses to undergraduate courses for students majoring in humanities studies.
- 2) in specialized computing courses using mockups tools

In all these courses, accessibility issues can be introduced from the beginning of the course and at the same time involve all the students, including those with special abilities, in a common project. Examples in this direction range from applying a methodology for searching, selecting and summarizing information focused on simplifying a daily life problem of students

experiencing difficulties, and to designing and developing an app or a device to help students with a motor disability to communicate, learn and use the educational material. Sound interface design principles can be introduced and applied using a project-based approach throughout the course and in individual students' learning paths. In addition, the authors recommend the Do-IT project (University of Washington, 2022) for guidance on accessibility issues as well as on how to embed HCI design principles into computing courses. The introduction of accessibility issues in computing courses is for the benefit of not only all computing learners but all users. Experience in these directions can be found in (Ross & Freeman, 2022) where students engage in developing accessible resources from first principles while learning to script web pages using both HTML and CSS.

Using abstraction, decomposition and a top-down approach design methodology are also useful when presenting and examining computer architecture. Visual block languages such as App Inventor and Snap! greatly help in this process by enabling details to be hidden that can be presented with an incremental spiral curriculum approach.

The integration of ADLCS into the P12 curriculum is in the early stage. However, the question asked at the ACM conference about the level of education required in order to study HCI (Churchill et al., 2013) is a daily challenge for all computing educators. When computing education concepts and skills are integrated into the entire P12 curriculum the questions that must be addressed are:

- 1) Do in-service teachers have the pedagogical, technological, and content knowledge in computing education to teach HCI principles appropriately and guide learners as they complete HCI related projects? Should this be the focus of the training? Or should we leverage on teacher mastery of pedagogical approaches and professional practices and leverage on the multidisciplinary experiences that can be found in every educational institution, like the schools?
- 2) What level of knowledge, skills and understanding should a P12 student have to be considered college and career ready?
- 3) In order to provide quality college and career ready P12 students, should computing education be a separate compulsory high school program?
- 4) How will equity in access sufficiently be addressed with students in lower socio-economic and/or rural areas that are unable to receive the necessary technology or applications to learn the content or skills?

As a final remark, we advocate an even stronger integration of computing curricula with other curricula starting from civic education and active citizenship with a particular emphasis on digital citizenship. In this respect we could mention, as examples, interdisciplinary topics such

as information search and selection, use data for sustaining conclusions and active civic participation, sustainable development. We can list many examples:

- 1) efficient hardware and software design and production
- 2) how recycling and efficient algorithms can contribute to sustainability and climate change by reducing the energy footprint of data centers and Information Technology systems and infrastructures
- 3) use techniques and approaches to scientifically evaluate websites, for information evaluation and fact checking

As a final thought, the tight integration between computer systems and HCI should be highlighted, brought into, and applied in all courses. The history of development of HCI interfaces can be taken as a proof of concept: we moved from textual interaction to graphical user interfaces and rapidly shifted to other user system interactions involving voice, language, and gesture recognition, with sophisticated artificial intelligence (AI) applications. Linking consideration to sustainable choices in developing both the hardware systems and the software infrastructures will give the possibility to add ethical considerations to all computing courses inside a curriculum. Greening considerations applies to all citizenships: from the enterprises and the researchers who build hardware and software systems to the users who utilize such systems. Efficiency considerations apply to building efficient computer architecture to building efficient AI applications for natural language processing and understanding used in modern user interfaces.

Table 3: A comparison of Computing Standards for Computer Systems and HCI

<p>Knowledge Categories</p>	<p>Required High School Competencies</p> <p>England Standards (DFE, 2013c) Italy (Nardelli, Forlizzi, Lodi, Lonati, Mirolo, Monga, Montresor, et al., 2017b), (M. E. Caspersen et al., 2022) Alabama, DLCS (Alabama State Department of Education, 2018a) USA (CSTA, 2017c), (CSTA, 2020c)</p>			
<p>Computer Architectures</p>	<p>Key Stage 3: Design, use and evaluate computational abstractions that model the state and behavior of real-world problems and physical systems</p> <p>Key Stage 3: Understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems.</p>	<p>Competencies:</p> <p>T-S-1 Understand the need to refer to the capabilities of an automatic executor in order to express algorithms in an unambiguous way</p>	<p>Abstraction. 1. Decompose problems into component parts, extract key information, and models to understand the levels of abstractions in complex systems. 2. Explain how computing systems are often integrated with other systems that may not be apparent to the user.</p> <p>Impact of Computing. 21. Explain how technology facilitates disruption of traditional institutions and services.</p> <p>Data. 32. Use data analysis tools and techniques to identify patterns in data representing complex systems.</p>	<p>CSTA Standard (CSTA, 2017c) 3A-CS-01-03. Explain and compare levels of abstraction of computing systems. Apply troubleshooting strategies.</p> <p>3B-CS-02: Illustrate ways computing systems implement logic, input, and output through hardware components.</p> <p>Teacher Standard 1B (CSTA, 2020c) Apply knowledge of computing systems. Apply knowledge of how hardware and software function to input, process, store, and output information within computing systems by analyzing interactions, designing projects, and troubleshooting problems.</p>
<p>Operating Systems</p>	<p>Key Stage 3: Design, use and evaluate computational abstractions that model the state and behavior of real-world problems and physical systems.</p> <p>Key Stage 3: Understand the hardware and software</p>	<p>Competencies:</p> <p>T-S-11. Recognizes the universal and multi-purpose nature of computer-based systems and understands the role of programs in transforming them into machines for</p>	<p>Systems. 34. Categorize the roles of operating system software. 35. Appraise the role of artificial intelligence in guiding software and physical systems.</p>	<p>CSTA Standard (CSTA, 2017c) 3B-CS-01 Categorize the roles of operating system software.</p>

	components that make up computer systems, and how they communicate with one another and with other systems.	specific purposes.		
Human Computer Interactions	<p>Key Stage 3: Design, use and evaluate computational abstractions that model the state and behavior of real-world problems and physical systems.</p> <p>Key Stage 3: Create, re-use, revise and re-purpose digital artifacts for a given audience, with attention to trustworthiness, design and usability.</p>	<p>Competences: T-S-12. understands the importance of user needs for the implementation of computer-based applications;</p> <p>Knowledge and skills: Digital Awareness O-S-N-2. To take into account the requirements of end-users in the implementation of computer-based applications.</p>	<p>Human/Computer Partnerships. 38. Systematically design and develop programs for broad audiences by incorporating feedback from users. 39. Identify a problem that cannot be solved by either humans or machines alone and discuss a solution for it by breaking the task down into sub-problems suited for a human or machine to accomplish.</p>	<p>CSTA Standard (CSTA, 2017c) 3A-AP-21: Evaluate and refine computational artifacts to make them more usable and accessible.</p> <p>Teacher Standard 1B (CSTA, 2020c) 4c. Design inclusive learning experiences. Use Universal Design for Learning (UDL), Culturally Relevant Pedagogy (CRP), and other techniques to support all students in successfully accessing and engaging with content.</p>

4.2 A Curriculum and a Learning Path for a First Course on Computing

In this section we present the design, development and evaluation of a first course on computing. In section 4.2.1 we present the design development and evaluation of the curriculum and in section 4.2.2 the design and development of learning paths around the presented curriculum.

4.2.1 Design, development and evaluation of a first course on computing.

Design principles

The curriculum aims to offer content and learning materials for a first course in computing suitable for all teachers and their students which, with adequate motivation, can be supported in climbing the learning pyramid from mere knowledge to creativity.

The fundamental ideas inspiring the curriculum are:

- 2) A low floor entry point suitable for all students and a high ceiling supporting the curiosity of all learners
- 3) Inquiry-based approach
- 4) Emphasis on design supported by many design tools
- 5) Different expressive registers
- 6) Block based languages supporting high cognitive skills
- 7) Many programming languages with a common interface
- 8) Many advanced topics
- 9) Multiple learning trajectories that can be personalized to the needs of each student
- 10) Interdisciplinary applications
- 11) Multiple delivery media, e.g. book, interactive ebook, online course, etc ...

In order to reach the low floor, high ceiling goal we envisage a cycle in the design process involving unplugged activities (Bell, et al., 2002), design tools such as flowgorithm (Cook, 2015), visual block languages and puzzles with an increasing level of difficulty, supporting students in their problem solving process.

The choice of using visual block languages leverages on the necessity, which has arisen from the rapid technological growth and exponential growth of the amount of available information, to sharpen the high order cognitive skills sought after by today's labor market (Manca, 2018). Visual block languages allow learners to focus on problem solving and high order cognitive skills, avoiding the necessity to acquire syntactical details required by textual languages. Those languages become necessary when other considerations, e.g. efficiency of execution, are of primary importance.

The learning material can be used for:

- 1) A first high school course on computing, e.g. for K9-K10 grade band (CASTA, 2016), (CSTA, 2017)
- 2) Pre-service teacher training without a major in computing
- 3) Teacher Professional Development (PD)
- 4) A first undergraduate course for students majoring in fields other than computing, e.g. the humanities.

The learning trajectories

Figure 1 depicts the main concepts in the curriculum and how they are linked. The concept maps can be navigated along many routes leaving the teachers the possibility to adapt the content to the class and each individual student.

We envisage a learning trajectory with a focus on developing CT. This will be produced by guiding students in acquiring a broad set of skills, useful not just to future computing professionals (Denning, 2017). The curriculum has the following strands:

- Computational thinking
- Digital literacy

- Soft and social skills

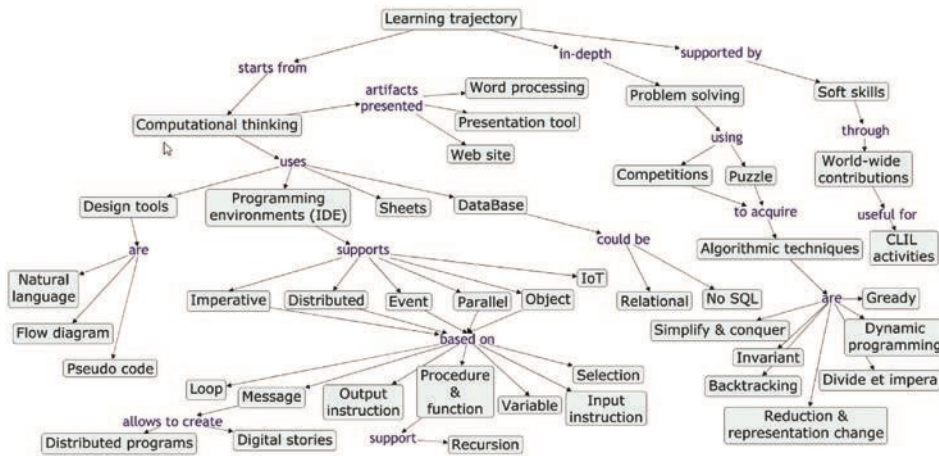


Figure 1. Curriculum concept map

The Computational Thinking strand

The CT strand uses a constructivist, student-centered approach grounded in cognitive theory/constructivism (Guzdial, 2018), and is based on the following activities:

- Reading, tracing, modifying and designing programs and algorithms expressed by means of:
 - Flow diagram (e.g. Flowgorithm)
 - Natural language
 - Pseudocode

Supported by activities requiring learners to translate from one representation to the others or to a visual block language

- Coding:
 - deluge of block languages, to experiment with core concepts in computing
 - translating the programs into a textual language
- Puzzle based learning:
 - algorithm design techniques: backtracking, divide and conquer, greedy, dynamic programming, invariant and so on.

The coding is supported by a deluge of block languages that, by sharing a common interface, allow teachers to leverage on their peculiar features to present and reason around core concepts in computing. Teachers can use the mutual support and reinforcement of the different programming and design tools, plugged and unplugged activities to offer a rich variety. For example, for parallelism unplugged activities such as the one proposed in (Bell, et al., 2015), (Tennessee Tech, s.d.) can support the plugged activities.

The author envisages in the curriculum the mutual support of plugged and unplugged activities, visual block-based and textual languages, multiple design tools to provide teachers and students with a richer set of design methodologies, tools, and expressive registers allowing each one to find the one most suited to her/his needs.

Table 1. A partial list of visual block languages used with a suggested progression and the key features of each language

	Block language	Key features
<input type="checkbox"/>	Scratch (Resnick, et al., 2009)	Easy to use. Movement, Pen, Control, Procedures
<input type="checkbox"/>	Scribble (Lane, Meyer, & Mullins, 2017)	Write on the stage. Create shapes
<input type="checkbox"/>	NetsBlox (Broll & Ledeczi, 2017)	Message with data. Distributed programming
<input type="checkbox"/>	Snap! (Harvey & Mönig, 2010)	Function. Recursion and functional programming. Parallel programming (e.g. map – reduce)
<input type="checkbox"/>	Tunely (Trower & Gray, 2015)	Multimedia data manipulation in one dimension
<input type="checkbox"/>	Pixly (Trower & Gray, 2015)	Multimedia data manipulation in two dimensions
<input type="checkbox"/>	App Inventor (Patton, Tissenbaum, & Harunani, 2019)	Event programming. Mobile app development. NoSQL database. IoT
<input type="checkbox"/>	Cellular (Lane, 2012),	Biological system simulation
<input type="checkbox"/>	Blop (Federici, Gola, & Ilardi, 2014)	Block language for C/C++. Step towards textual languages
<input type="checkbox"/>	BlockPy (Bart, Tibau, Tilevich, Shaffer, & Kafura, 2017)	Data manipulation. Automatic translation in Python
<input type="checkbox"/>	Edgy (Cox, Bird, & Meyer, 2017)	Data structures. Bridge between unplugged and plugged
<input type="checkbox"/>	GP (Monig, Ohshima, & Maloney, 2015)	Multimedia manipulation. Introduction to class without inheritance
<input type="checkbox"/>	Parallel programming (Feng, Gardner, & Feng, 2017)	Blocks for parallel execution

The puzzle-based approach is a leitmotiv of the whole curriculum with puzzles proposed in all chapters and modules. Table 2 lists some of the major algorithm techniques and the puzzles used to introduce them. All the CT, and puzzle activities in the same module and across the whole curriculum, shows a progression from core to intermediate and advanced with a clear indication provided in the companion teacher's book. For students, an icon indication can guide them in choosing the preferred activities. The progression is supported by clear and sharp classification and progression provided in (Levitin & Levitin, 2011).

Table 2. Algorithmic techniques with some examples of puzzles proposed in the curriculum

Algorithmic techniques	Puzzle
Greedy	Pearson, bridge crossing and lamps; Huffman code
Decrease and conquer	A fake among eight coins, fake coin detection with a spring scale;
Divide and conquer	Tromino puzzle, 2n counters in a nxn board
Change of representation	Two jealous husbands, Stack of fake coins, Drawing a figure without lifting the pen; sequence of words
Dynamic programming	Shortest path counting; Knapsack problem; Common subsequence, Palindrome counting
Invariant	Break a chocolate bar; Colour of last marble; Knight movements; domino and tetromino tiling
Inference	Sequence of facts and conclusion;
Backtracking	Four and n queens; CriptoAlgorithms & CriptoArithmetica
Induction, proof of correctness	Knapsack problem, divide a rect angle in triangles

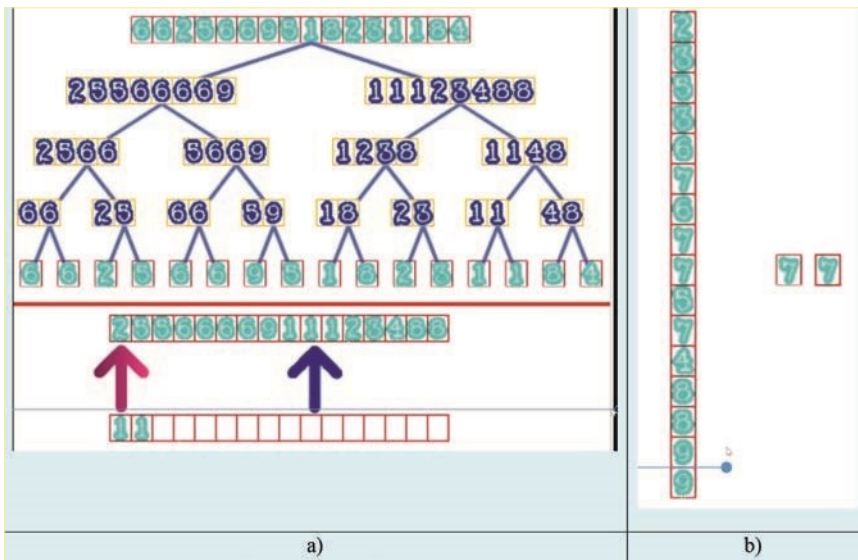


Figure 2. Sorting algorithm animations: a) Merge sort; b) Bubble sort

An inquiry-based approach is used to give the students a central role. Figure 2 shows a snapshot of two animations of merge-sort and bubble-sort. Students are requested to watch the animation and, before any educational intervention, are guided by a set of questions in discovering the algorithms behind this sorting processes. The set of questions goes into deeper detail in successive runs, e.g. midterm and final.

A similar approach is useful for algorithmic techniques such as backtracking. Figure 3 shows an example of a graph created with Edgy and its topological order.

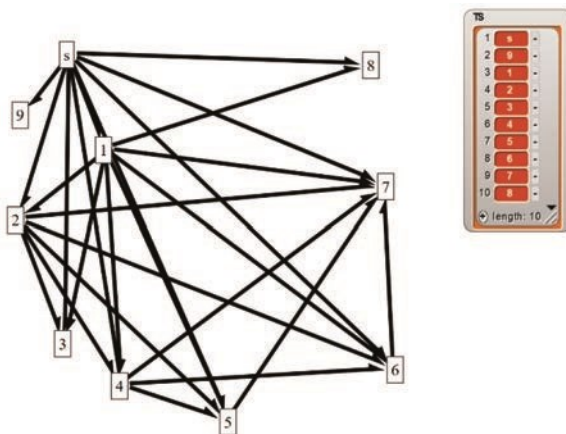


Figure 3. A graph and its topological order obtained with Edgy

The Digital Literacy strand

The digital literacy strand covers the following topics:

- Conduct bibliographic research.
 - Being able to search, select, summarize, visualize and reference quality information.
- Particular emphasis is given to a rigorous process with clear and objective indications

for every step: from selecting the search engine to selecting the best key phrases, for judging the source of information, verifying it and so on.

- Office automation. The major suites for office automation are presented, both proprietary, such as Microsoft, and open source such as LibreOffice and OpenOffice. Emphasis has been given to online and cloud-based tools as a way to hone collaboration and group work skills. To present the suites, an explorative approach is suggested, asking the students to find ways to accomplish tasks, either by exploration of the interface or by searching through the technical documentation. This is the best way to cope with different interfaces changing over device, over software and over time. The explorative approach is always preferred and the correct solution, e.g. the sequence of steps to accomplish the task is given at the end of the activities, frequently only in the companion teacher's guide. Interface design principles are given by comparing the different interfaces available in the different devices (desktops, tablets and smartphones) and by analysing commonalities both intra applications inside the same family of software tools and inter office suites.
- Particular emphasis is given to searching, retrieving, analysing, visualizing and storing data. The importance of open and linked data is used as the key starting idea. Data are searched and retrieved and then analysed and visualized using Excel, Libre and Google sheets.
- Finally, storing data in databases (both relational and NOSQL) is considered. Activities for designing and querying a relational database and ways to visualize the data via an ad hoc designed interface are presented and suggested. The difference with a NOSQL database are explored and practical mobile applications are designed and developed by means of App Inventor and available NOSQL database components.

The soft and social skills strand

The importance of soft skills as well as social skills is recognized worldwide. For this reason, these topics are discussed through contributions from leading experts to open a window onto the world for students, giving them the possibility to compare the experiences from different countries and cultures. Among the topics covered, to cite just a few, it is possible to recall:

- Professional ethics
- Informal education
- Humanitarian Free and Open Source Software (HFOOS) – Free and Open Source Software (FOSS) (Hislop, Jackson, & Ellis, 2015), (Morelli, et al., 2009)
- Computer Science and its impact on society
- Inclusive education
- Mens sana in corpore sano (Healthy brain in healthy bod). Importance of sport
- Sustainable development

- Technologies and well-being

Contributions come from leading experts from: Australia; Canada; Europe: England, Ireland, Italy, Lithuania, Spain, Switzerland; New Zealand and USA working in universities, international organizations, international institutions, enterprises.

This contribution can be used as Content Language Integrated Learning (CLIL) activities for students learning English as a second language.

Evaluation of results and discussion

The content derives from several experiences described and qualitatively and quantitatively evaluated in different studies (Giordano & Maiorana, 2014), (Giordano & Maiorana, 2015), (Maiorana, 2019). The positive effects of a first version of the curriculum have been evaluated by means of student progress on assessment evaluation and student survey (Giordano & Maiorana, 2015). Starting from the 2013 academic year, the curriculum was iteratively designed, developed, deployed, evaluated and improved. Each year the curriculum was field-tested in at least one class with an average of 25 students. Students, majoring in CS, were in either the first or second year (K9 or K10) of an Italian high school. The average female population was 15%. An average of 15% of students with disadvantaged socioeconomic status can be estimated. K9 students approached the course without mandatory prerequisites. For K10 students, a mandatory knowledge of basic problem-solving techniques and major programming constructs in an imperative language including procedures and functions were required. On average in each class, there were two students with learning disabilities (dyslexia or dysgraphia) and one student with special education needs. Curriculum effectiveness was qualitatively evaluated through student surveys and pre-test post-test assessment. When possible, comparisons with other classes in the same school taught by different professors were performed. The main conclusion that can be drawn from the evaluation process is that overall 14/16 years old students at the beginning of the course tend to underestimate block languages, considering them too simple, useful for younger people, not teenagers. As the progression of the topic becomes tougher and challenges the students, their appreciation of block languages increases since these languages allow the students to easily reason on the problems, construct artifacts and test them without worrying about too many details (Giordano & Maiorana, 2014).

Teacher feedback was obtained from five anonymous teacher reviews regarding the curriculum. The reviewers were located in Italy and the reviews were collected from mid 2017 to mid 2018. Other feedback was obtained from direct observations, informal unstructured teacher interviews inside a pre-service and professional teacher development course run in 2015. The teacher development course was attended by 40 teachers. Thanks to a Google CS4HS grant, the project ran a teacher workshop where by means of surveys, and meeting with teachers the author obtained feedback about learning resources, teachers' needs, and

expectations, and features desired for a curriculum. Analysing the teachers' feedback, it is possible to summarize the following key ideas:

- 1) On first impression, the quality of the proposed material and the diversity of the materials seem to disorientate some of them. For this purpose, indications of different progressions and a teacher guide offer a way to get acquainted with the curriculum. This guide can be used just as an ice-breaker; the experience and teachers' knowledge of their students will allow them to navigate the curriculum and find the best activity suited for the next steps in the zone of "Proximal development" for each individual student.
- 2) The ample diversity of communication channels and expressive registers, tools and technologies coupled with clearly stated progression and levels of difficulties allows for an inclusive and equitable approach. This approach is strengthened by an attention to learners with special abilities (UNESCO, 2017) in content delivery (edX, 2019).
- 3) The teaching approach sustained by inquiry-based pedagogies (Hazelkorn E., et al., 2015), Peer Instruction (Porter, et al., 2016), (Peer Instruction, 2019) and Process-Oriented Guided Inquiry learning (Education ambivalence, 2010), (Computer Science POGIL, s.d.) has the advantage of giving students an active role. By flipping the classroom (Bishop, Verleger, & others, 2013), (Karabulutllgu, Jaramillo Cherez, & Jahren, 2018) teacher-led and peer-led classroom time can be focused on problem-solving activities. Solving puzzles, engaging in projects (Blumenfeld, et al., 1991) and realizing artifacts to solve real world problems (Wolber, 2011), alone, in pairs and in groups allows learners to hone their collaboration and communication skills (Griffin & Care, 2014).
- 4) The interdisciplinary approach seems to be a promising way to expose students to computing, especially in school streams (e.g. classical studies) where computing is not a mandatory topic. In this case, where there is a lack of teachers with a specific certification in computing, approaching computing with applications in the teachers' and student's comfort zones represents a low floor entry point.
- 5) Use of formative assessment (Giordano D., et al., 2015), (Oates, Coe, Peyton Jones, Scratcherd, & Woodhead, 2016) supported by the above-mentioned pedagogies greatly supports students' activities and teachers' instructional process.

Undergraduate students with a major in the Humanities (Maiorana F., Computational Thinking and Humanities, 2018), most of them exposed for the first time to computing, reported, after overcoming foreseeable difficulties, joy and fulfillment in developing real work applications related to their subject of study and future profession and appreciated the design methodologies, the block language (Patton, Tissenbaum, & Harunani, 2019) and the possibilities to create mobile apps and sites showcasing their project portfolio.

4.3.2 Learning trajectories

From the above analysis, it is evident that there is a pressing need both to extend the introduction of computing instruction to all students and to improve students' success rates and the quality of teaching and learning of introductory programming.

Contribution: This work will explore, analyze and compare three different learning trajectories, namely variable first, considered as the baseline approach, loop first, and function first, to empower students' development of 21st century computing competencies (Clear, 2020). All the three approaches are guided by relevant research strategies (Maiorana, 2019), such as the role of variables (Hosanee, 2018) and types of loops (Arnou, 1994). The collected data suggest that a function first approach favors students' ability in designing a program and dividing into procedures and functions, giving more time to the learning of problem solving and algorithm design strategies expressly introduced in the curriculum at an early stage (Maiorana, 2019), (Levitin, 2017).

Outline: A brief introduction on the data collection process used, and the multiyear teaching experience related to the learning trajectory will be presented. The teaching experience in introductory programming courses both at the high school and at the undergraduate level related to the three different learning trajectories will then be presented starting from function first, then loop first, and finally variable first, along with the suggested class schedule for an introductory programming course. At the end of each section the learning trajectory will be evaluated, and the main lesson learned and best practices will be presented. A final section will report conclusions and suggestions for further work.

The data collection process

The learning trajectory has been evaluated using data collected by:

- a) in a high school setting:
 - a. by analyzing and comparing nine written formal assessment tests and the relative grades each school year. The written formal assessment tests were evaluated by the author of the work and verified by another colleague co-teaching each course.
Difference in grades were resolved among the two instructors
 - b. by analyzing and comparing four written formal lab assessment tests and the relative grades each school year. The written formal lab assessment tests were evaluated by the co-teacher of each course and verified by the author of the course. Difference in grades were resolved among the two instructors

- c. by analyzing and comparing notes and grades on six formal oral assessment tests each year. The assessment activities were conducted by one of the teachers and verified by the other teacher
 - d. the final midterm and end of year grades in the initial programming course taught by the author.
- b) in the undergraduate initial programming course
- a. by the written, lab and oral formal assessment activities undertaken by each student at the end of the course. The assessment was evaluated by two instructors of the course and by a teaching assistant

Learning trajectories for a first course in computing

This work will present some learning trajectories for the Computational Thinking strands of the curriculum presented in (Maiorana, 2019) an enacted version (Falkner, 2019a) of the intended curriculum highlighted in the Italian proposal (Nardelli, 2017), (Forlizzi, 2018). This work will describe three different learning trajectories, report on case studies relative to their application and synthesize the main lesson learned from each experience, namely

- 1) Function first
- 2) Loop first
- 3) Variable first

The learning trajectory focusing on a variable first approach covering output instruction, variables, input instructions, loops, arrays and matrices, procedures and functions and sorting and searching algorithms will be used as a reference learning trajectory. The results of the other two learning trajectories will be compared against it.

Function first

Abstraction and decomposition of a large problem into a smaller one represents the major challenge in designing complex software and a key aspect for a successful educational process in computing. This learning trajectory offers the possibility to focus from an early stage of a first computing course on the decomposition aspects, leaving the possibility to introduce advanced topics like recursion (Maiorana, 2020a), (Maiorana, 2020b) typically taught at the end of a second computing course (CS2 course), at an early stage of the educational path. This has been successfully proposed to students as an elective activity by requiring them to design and develop both an iterative and recursive version of the functions. Games like Lightbots (Aedo, 2016) have been used to foster discussion among learners and informal competitions to find the best solutions in terms of number of moves. Other games like CargoBot (Tessler, 2013), (Lee, 2014) move in the same direction. Table 1 reports a class schedule template and the associated competencies that can be used for a 4-month

undergraduate term. For a first course in high school at the K10-K12 level, the content can be used in a two-term sequence leaving more space for project based activities and assessment.

Table 1: A class schedule template for a procedure first learning trajectory

Description	Computing concepts & Competencies	Time
Procedure and function	Function	1 month
Receive input	Input & variables	
Receive and check the correctness of input	Conditions and logic	
Draw simple figures	Loops & recursion	2 weeks
Draw complex figures	Libraries	1 month
Receive and check the correctness of input for complex types	Array & matrices	1 month
Write a library to manipulate different types of data	Date, time, strings	
Manipulate variables, e.g. swap 2, sort 2, 3, 4, invert	Deepening variables	2 weeks
Manipulate list of data, e.g. sort, swap, check for a condition (e.g. is sorted), find elements with a property (e.g., minimum), manipulate multiple list (e.g. merge)	Use of list of data in one or multiple dimensions	2 months

When functions are presented at the beginning, it is possible to start from simple programming like the one presented in Table 2. In the assignment it is possible to use three different levels of difficulties. In a high school setting this will allow instructors to use the first assignment as a minimum requirement for passing; the second one for a good grade; and the third one to challenge the students aspiring for the best grades and wishing to major in computing in their further undergraduate studies. The above described approach can be used as a template for all the assignments, allowing teachers to adapt the template to the needs of each student.

Table 2: An example of assignment for a function first learning trajectory

1) Write a program that draws the following figure. Use at least two procedures with the appropriate parameters: one to draw a rectangle (start with this one) and one to draw a triangle.

2) Repeat the figure by:

- adding a line to each geometric figure.
- shifting the figure to the right by a given number of characters
- assuming the figure start at the rightmost top edge of a canvas shift it down by an appropriate number of lines

3) As an elective activity, find a way to draw a version of the same figure rotated by 45 degrees

```

* * * * *
* * * * *
* * * * *
* * * * *
*
* * *
* * * *
* * * * *
* * * * *
* * * * *
* * * * *

```

A whole functional curriculum for middle school students³ has been developed as a spin off of the Beauty and Joy of Computing curriculum using the Snap! block-based programming language (Garcia, 2012), (Harvey, 2017).

Learning trajectory evaluation, lessons learned and best practices

The learning trajectory was experimented at grade 11 out of 13 in an Italian high school on a two-term course through assessment results, in-class student survey and informal interview. The average class size was of 25 students with the majority (80%) being male. The students joined the course having already taken two general courses composed of three-hour lessons over a thirtyweek period. In the first course they learned digital literacy and in the second they learned information technologies and were exposed to some programming concepts not covering functions and procedure for one 5-week module (15 hours). The course lasted 30 weeks and the last 8 weeks of the course covered an introduction to basic algorithm design techniques and students were engaged in project design and development. The programming language used in courses at this grade level was Java.

Regarding assessment, the average grade for questions related to functions, procedures, decomposition, and abstraction was 1.12 grade point average (GPA) above the GPA of the

reference course with a variable first approach. The GPA was computed on a scale from 1 to 10. The students liked the approach. The comments from the students were variegated and ranged from

- “I was supported by the approach that allowed me to master procedures and functions” from a (male) student who was repeating the 11th school year since he had not passed in the previous year; to this:
- “I liked the approach at the beginning, it was simple and effective, in the long term I really enjoyed the challenging aspects that kept my interest alive” from one of the most brilliant students (female).

The approach was also used at grade 11 out of 13 in an Italian high school. At this grade students, after completing the course at grade 10, must focus on object-oriented design and programming and data structures. The class schedule reported in Table 1 was covered in one month using a different programming language, either C# or C++. Table 3 reports an example of a typical first programming assignment.

Table 3: An example of an assignment for a function first learning trajectory suitable for a second level computing course

Write a procedure or function that:

1) receives a text string as a parameter and displays this string on the monitor.

Write the main program to test the procedure. 2) receive as a parameter:

- i. a text string
- ii. a minimum value of integer type
- iii. a maximum value of integer type

display the text string on the monitor, then the minimum value and the maximum value, receive

an integer value between the minimum value and the maximum value and return the read value. Write the main program to test the function.

3) exchanges two received values as a parameter. Prepare appropriate tests.

4) receives an integer type parameter and increments the value of the parameter. Check that the function actually increments the value and avoids overflow issues. Try to write the same function recursively.

5) Write a function that uses only the procedure written in the previous step to sum two integers received as a parameter. The function calculates the sum of two numbers using only the increment and assignment. Try to write the same function recursively.

According to a recent study, data collected through students' formative assessment aiming at guiding reflection on animations related to iterative (bubble sort) and recursive sorting algorithms (merge sort) tend to perceive procedure, functions, and recursion as simpler. The curriculum was evaluated with last year undergraduate students majoring in humanities attending their first course in programming and with younger students attending a first programming course. Initial studies showed comparable performance of student comprehension of iterative and recursive sorting algorithm (Maiorana, 2020 a) and further research is necessary in this direction. International experiences such as the Beauty and Joy of Computing middle school curriculum³ where, in the first unit lasting six months, a complete functional approach is presented to middle school students, goes in the direction suggested in (Maiorana, 2020a).

Lesson learned

The main lesson learned is that this learning trajectory has a steep learning curve at the beginning and requires a lot of effort to sustain the students, but in the long term it produces the best results in terms of students learning of procedures, functions, decomposition and abstraction, and to some extent algorithm design, which represents the key concepts that students must acquire in an initial programming course.

Loop first

Using this learning trajectory, students start using simple loops (Astrachan, 1998), such as repeat a fixed amount of time, to draw figures with an increasing level of difficulty (Maiorana, 2019). The main focus has to be on identifying loop invariants and their relationship to pre and post conditions (Arnou, 1994), (Back, 2009), (Furia, 2014), (Ginat, 1995), (Tan,1992). Scratch, Snap! and their dialects as well as App Inventor are suggested as main programming languages. The learning trajectory starts from drawing simple geometric figures using loops repeating a fixed amount of times. The use of different types of loop shown in Figure 1 is explored starting from translating the same program using different types of loops. Drawing figures gives ample freedom to learners to design their preferred drawing suggesting the composition of a simple figure. Messages in development environments like NetsBlox (Broll, 2018) allow for;

- 1) in the realm of storytelling, managing stories with multiple paths and exploring data structures such as finite state machines,
- 2) develop networked games granting an easy transition to message with data, remote procedure call and hence procedure and functions.

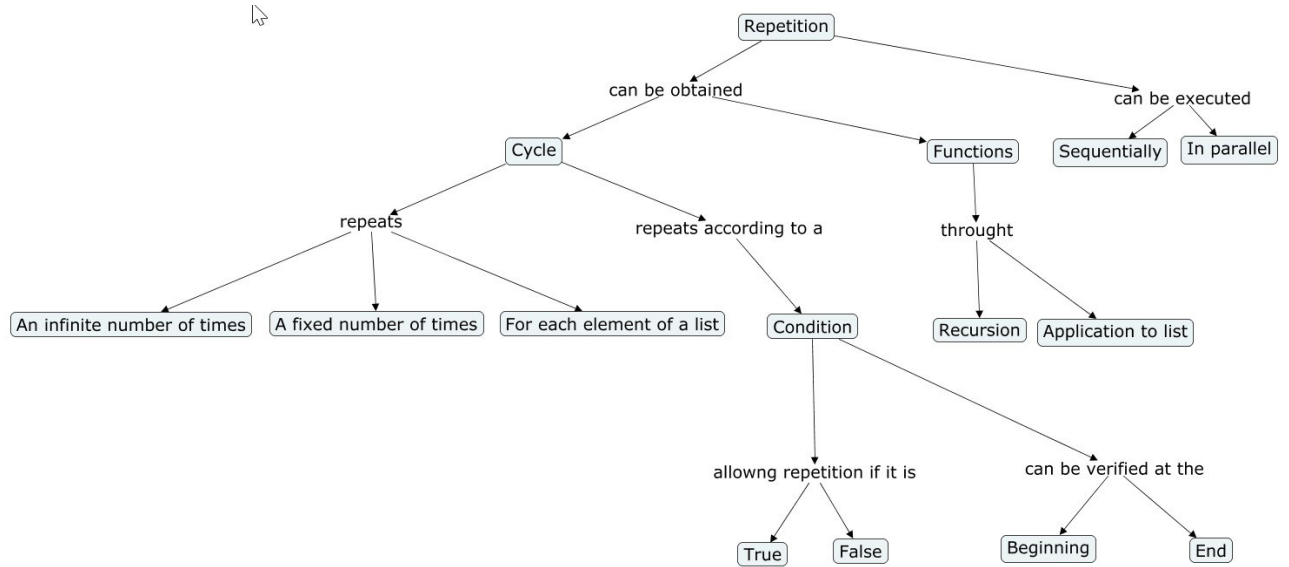


Figure 1: Different types of loops

Procedure and functions are then used to construct complex figures starting from refactoring solutions developed using nested loops or decomposition obtained through message without and with data. From here, ample space is given to student creativity. Exploring variables and their roles starting from receiving single and multiple input and performing basic algorithms on lists in one or multiple dimensions complete the learning trajectory which is summarized in Table 4.

Table 4: A class schedule template for a loop first learning trajectory

Description	Computing concepts & Competencies	Time
Draw simple figures	Loops	1 month
Digital stories, conditions, and messages	Conditions, message passing without and with parameters	2 weeks
Procedure and function	Function	2 weeks
Draw complex figures	Loops, procedures, functions, and recursion	1 month
Variables and their role	Variables	2 weeks
Receive input and check their correctness	Input & variables	2 weeks
Receive and check the correctness of input for complex type	Arrays & matrices	2 weeks

Manipulate variables, e.g. swap 2, sort 2, 3, 4, invert	Deepening variables	2 weeks
Manipulate list of data, e.g. sort, swap, check for a condition (e.g. is sorted), find elements with a property (e.g., minimum), manipulate multiple list (e.g. merge)	Use of list of data in one or multiple dimensions	1 month

Learning trajectory evaluation, lessons learned and best practices

The learning trajectory was evaluated in K9 and K10 courses out of 13. The class ran for a whole year, i.e. 30 weeks. The learning trajectory in the realm of K9 and K10 courses was designed and developed over the course of a decade of experiences described in different studies (Maiorana, 2014), (Maiorana, 2015), (Maiorana, 2019). These experiences were qualitatively and quantitatively evaluated, showing a good appreciation of the whole approach. Starting from loops and drawing tasks of geometric figures allows student to easily compare the program output and the expected result allowing them to better acquire debugging competencies. Finally a loop first approach was evaluated in an informal one-year project involving pre-service and in-service teachers and their students, in a project supported by a Google Computer Science for High Schools grant, using the App Inventor block based language (Gray, 2012), (Hoffman, 2019), (Patton, 2019). The approach was positively evaluated by the majority of the teachers (> 85 %) with an increase in self-evaluation in the post survey compared to the pre intervention survey.

Variable first

With a variable first approach it is possible to make a learning trajectory built around variables and their use. Variables can be used in different roles² (Sajaniemi, 2005) (Hosanee, 2018). The simplest role for a variable is to store a value that does not change during the program. Some languages use constants to store values that do not change during program execution. Other roles according to (Sajaniemi, 2005) are counter, accumulator, recent value, extreme value, follower, flag, temporary and index and are summarized in Table 5 which provides an example for each role developed using flowchart as design tool (Cook, 2015). The table can be used as a progression. A variable first approach is used around the world and in this work it is considered the reference learning trajectory. The learning trajectory is summarized in Table 6.

Table 6: A class schedule template for a variable first learning trajectory

Description	Computing concepts & Competencies	Time
Variables and their properties, operators, expression.	Variables	2 weeks
Conditions and logical operators	Conditions	2 weeks
Receive input and check their correctness	Input & variables	2 weeks
Manipulate variables, e.g. swap 2, sort 2, 3, 4, invert	Deepening variables	2 weeks
Type of Loops	Loops	1 month
Manage complex data type, e.g. date and time	Structures	2 weeks
Manage strings		2 weeks
Manage containers	Array and Matrices	2 weeks
Procedure and function	Function	2 weeks
Manipulate list of data, e.g. sort, swap, check for a condition (e.g. is sorted), find elements with a property (e.g., minimum), manipulate multiple list (e.g. merge)	Use of list of data in one or multiple dimension	1 month

Learning trajectory evaluation, lessons learned and best practices

The learning trajectory was evaluated during five consecutive years from 2007 to 2012. During these years the suggested class schedule in Table 6 was implemented with several pedagogical approaches rooted in inquiry-based and student-centered educational methods based on constructionism and project-based activities. Overall, the approach is best suited for a bottom up approach: details first up to decomposition and abstraction. According to multiple-year evaluation of student formal assessment, this approach leads to worse results in terms of function and decomposition issues. This is since students have less time to practice functions and their use and have less exposure to decomposition. The learning curve is faster and leads to better results at the beginning of the formal assessment, but these better average results apply only to syntactic aspects and are related to Knowledge or low levels in the Bloom taxonomy (Bloom, 1984), (Anderson, 2001), (Adesoji, 2018). In the long

run, at the end of the school year students present, as stated before, a poorer performance in concepts related to functions and decomposition.

Comparison of the learning trajectories

The function first learning trajectory and the loop first learning trajectory are compared with the variable first learning trajectory which is considered as the reference learning trajectory. The variable first approach has revealed the worst result in terms of student performance evaluated through grades and assessing the five overlapping domains that have to be mastered by a good programmer (du Boulay, 1989), namely: 1) general orientation, 2) notational machine; 3) syntax and semantics; 4) structure and use of a plan; 5) pragmatic skills like planning, developing, testing, and debugging. This result is in line with the main conclusion on students' misconceptions reporting a lot of issues related to variables, starting from conceptual knowledge (Qian, 2017).

The loop first approach revealed better overall student engagement for all the students in a typical bimodal class (Guzdial, 2007), i.e. both the 20 % top performers and all the others. Students are more engaged in developing programs since the programs exploit what computers do better, i.e., repeating operations. The approach presents a lower entry point especially in block-based programming environments allowing for an immediate visualization of the effect of the program on the action performed by the sprites on the stage. The conclusion is in line with recent studies reporting minimized students' misconceptions on loops using block based programming languages (Mladenović, 2018).

The function first approach revealed better student performance in all the five overlapping domains allowing an early introduction to threshold concepts (Meyer, 2003) such as procedure, functions, program decompositions and abstraction. This learning trajectory allows for an early introduction of algorithm design strategies and problem-solving strategies. The suggested approach is to couple a puzzle-based approach (Maiorana, 2019), (Levitin, 2012) with an implementation and testing phase in a programming environment. The last activity can be supported earlier in the curriculum with a function first approach. The approach, like the loop first approach, is also highly motivating and engaging for students. The minor drawbacks of requiring a more demanding effort from students can be avoided by an appropriate pedagogical approach, such as the suggested inquiry based approach, and appropriate technologies, such as block based languages.

4.3 A learning path for computing and civic education

According to the ideas presented and discussed at the panel (Maiorana, Quille, et al., 2022) moderated at the EDUCON 2022⁶⁴, where the presenters, by leveraging on the experience as learning resource developers (Becker, 2022), (Maiorana, 2019), (Maiorana, 2021), (Nolan, 2019), it has emerged the necessity to construct learning paths for guiding students as responsible citizens through resources connected with relatable contexts to student's interest and develop learning paths, deployed in learning resources, from problems to projects, in the K12 domain.

The relevance arises from international studies verifying the intended student's competencies which provide evidence to support the need to improve the students' competencies related to computing and digital citizenship. The Programme for International Student Assessment will assess for the first time Creative Thinking (Pisa, 2021), the International Civic and Citizenship Education Study (Schulz, 2018), scheduled for 2022, have focused on civic and citizenship education from societal context with an emphasis on the United Nation (UN) Sustainable goals (Desa, 2016) and education target. These studies report the need to improve students' competencies in computing and responsible citizenship which must be related to enacted educators' actions engaged in both formal and outreach.

Supported by the previous discussion a study has undertaken with the aims:

- 1) To obtain a picture of Italian students' strengths and weaknesses in three different areas: Computer and Information Literacy, Civic and Citizenship Education, Collaborative Problem Solving, and their learning evolution from 2015 to 2018.
- 2) To obtain a picture of teachers' enacted actions and outreach activities related to computing and coding and their evolution from 2014 to 2022.
- 3) To identify core competencies and commonalities in international curricula in 3 different areas: Computer and Information Literacy, Civic and Citizenship Education, and Computer Science.
- 4) To identify a learning path that, by using data as a glue, fosters the development of students' competencies in the above-mentioned areas.

According to the analysis presented in (Maiorana, under review) related to the Italian students' strengths and weaknesses in three different areas: Computer and Information Literacy, Civic and Citizenship Education, Collaborative Problem Solving in the following

- a) section 4.3.1 draws a pictures of activities related to Coding, Computational and algorithm thinking in Italy
- b) section 4.3.2 discusses on competencies in computer and information literacy, civic and citizenship education and computer science

⁶⁴ <https://educon-conference.org/educon2022/round-tables.php>

c) section 4.3.3 presents a learning path related to the above domains

d) section 4.3.4 synthesizes the main discussion topics

Furthermore the importance of social science theories in education will be further discussed in section 4.3.5.

4.3.1 Outreach activities related to Coding, Computational and Algorithm thinking

In Italy a strong effort has been made to introduce computing as a compulsory subject at least for all mandatory grade levels (Forlizzi et al., 2018), (Caspersen et al., 2022). Despite this effort, a compulsory curriculum for all grade levels from pre-primary to high school is still missing. The same applies at the undergraduate and graduate level up to master courses where a formal course in computing is present in almost all programs but the lack of intended curriculum guidelines produces a very broad variety with courses still mainly based on office automation content up to a more sophisticated course covering aspects of information technology, computing coupled with pedagogies and technologies best suited to teach the content. Unfortunately, this impacts teacher preparation starting from students aspiring to teach in primary schools. At the non-formal level, a lot of initiatives have been carried out in Italy, e.g., brebas task (Lonati, 2020), Problem Solving Olympics (Borchia et al., 2018), Italian Olympiads in Informatics (Italiani, 2011), (Amaroli et al., 2018), Programming the future (Corradini & Nardelli, 2021).

The hour of code (Yauney, 2022) and the code week initiatives have fostered outreach and formal coding initiatives across the world and across Europe (Moreno-León & Robles, 2015) in a playful environment (Klopfenstein et al., 2018), (Klopfenstein et al., 2019), (Bogliolo et al., 2021). In particular, the EU Code Week initiative has registered a steady increase in initiatives with Italy in a leading role. Tab. 1 reports the data on the number of events organized in Italy; positions are listed according to the number of events per population and the total number of events, according to the official website data.

Tab. 1 - Code week events⁶⁵

Year	# Events	Position	Total # of events
2014	333	18	318
2015	2346	5	7573
2016	14320	2	22964
2017	16534	2	24912
2018	20223	2	43524
2019	17529	5	71167
2020	9730	9	68415
2021	17913	7	78469

⁶⁵ <https://codeweek.eu/scoreboard>

4.3.2 Competencies in Computer and Information Literacy, Civic and Citizenship Education, and Computer Science

From the International Computer and Information Literacy Study (ICILS 2018) (Fraillon et al., 2019), (Fraillon et al., 2020a), (Mikheeva & Meyer, 2020), the following technology-mediated educational priorities for middle-school students relevant for this paper can be summarized:

- finding and synthesizing relevant resources,
- connecting to people and networks,
- knowing how to present and express oneself online and through online systems.

According to the International Civic and Citizenship Education Study ICCS 2016 (Schulz et al., 2016), (Schulz et al., 2018), (Köhler et al., 2018), (Losito et al., 2018), the following new areas can be highlighted:

- Environmental sustainability in civic and citizenship education
- Social interaction at school
- The use of new social media for civic engagement
- Economic awareness as an aspect of citizenship (see PISA 2022 financial literacy framework)
- The role of morality in civic and citizenship education

According to the PISA 2015 Collaborative Problem Solving (OECD, 2017), (Peña-López, 2017) the competencies required are:

- establishing and maintaining shared understanding;
- taking appropriate action to solve the problem;
- establishing and maintaining team organization

to which the competencies related to individual problem solving identified in the PISA 2012 framework (Peña-López, 2012) should be added, namely:

- exploring and understanding;
- representing and formulating;
- planning and executing;
- monitoring and reflecting.

According to the recent global competence framework, on students' ability to interact with the wider world around them, assessed in the Programme for International Student Assessment (OECD, 2019), (Ramos, 2016a), (Ramos, 2016b) students have to equip themselves with global competencies in four areas:

- examining issues;
- understanding perspectives;
- interacting across culture;
- taking actions

involving students' development encompassing cognitive, socio-emotional and civic development. In the recent publication (Boix Mansilla & Schleicher, 2022), the authors stress the importance of education in helping students build

- curiosity, i.e., opening minds;
- compassion, i.e., opening hearts;
- courage, i.e., mobilizing cognitive, social, and emotional resources to take actions.

In this way education becomes the “best weapons against the biggest threats of our times: ignorance – the closed mind; hate – the closed heart; and fear – the enemy of agency”. The work provides pedagogical principles and case studies, ranging from math to Content and Language Integrated Learning (CLIL), from formal to non-formal activities, to illustrate the guiding principles for shifting from “accumulating information” to “thinking with and applying information in a range of novel, often ill-defined situations”. Equipping students with these global competencies ensuring an inclusive approach granting a quality education for all is considered the only means to drastically reduce the number of NEETs.

4.3.3 Learning path

In this section, by using data as glue, we will propose a learning path to foster the development of students' competencies using curricula and learning resource development in the domains of Computer and Information Literacy, Civic and Citizenship Education, and Computer Science. Leveraging the studies presented in section 2, five learning resources were developed inside a whole curriculum around digital citizenship⁶⁶. The authors developed five learning objects aimed at guiding teachers and their students on:

1. Wise use of time, in particular when navigating the Internet.
2. Ecological footprint of information and communication technologies.
3. Data, a precious resource in the information age.
4. The importance of kindness in real life and in the virtual world over the internet.
5. Fact-checking and verification of information sources.

Providing resources and tools for constructing the learning path that could span from one lesson to months of activities.

The learning resources were designed with the following design principles:

1. Leverage reviews and state of the art in research
2. Use case approach: adapt the same topic and learning path to different audiences:
 0. From kindergarten to high school.
 1. Formal and non-formal education.
3. Plan to evaluate the educational activities with a pre-test and post-test approach
4. Rich set of student-centered assessment activities, both formative and summative

⁶⁶ <https://it.pearson.com/pearson-scienze/corsi-secondaria-2-grado/consapevoli-in-rete.html>

5. Provide solutions with a focus on the process rather than on the solution
6. A rich set of references and web links

The key reflection from the work can be summarized in:

1. (linked) open data is a key enabler for active citizenship (not only digital).
2. The importance of searching for, selecting, analyzing, interpreting, and presenting conclusions grounded in quality data is a key process to develop global competencies and big picture thinking.
3. The importance of student-led teacher-guided reflection on the process to conduct the previous steps.

The learning resources will be evaluated during the following school years, aiming to involve networks and professional communities of practice (Maiorana, 2020), (Maiorana, 2022).

4.3.4 Discussion

From the analysis of the large-scale international studies report, the international instrument to ascertain enacted practices, non-formal Europe-wide outreach activities, the authors present the following reflections which arose from the proof of concept experience report in developing learning resources targeting teachers and their students at the national level:

1. Shared reflections on intended and enacted curricula and a research-based approach to improving daily teaching practices are essential to better design learning resources and educational activities.
2. Netiquette and respect are indispensable rules for communication and democracy.
3. Data is an indispensable resource for democracy.

In the long run, the ultimate goal of each and every educational action is to reduce the number of people neither in employment nor education and training. Besides providing inclusive access to quality education for all, actions must be taken to support students in their learning path after compulsory education. Leveraging previous international experiences in introducing computing in the USA (Morelli, 2014) where teachers received a stipend to attend a summer professional development course on computing and then taught the subject in the following school year, we advocate for financing, at the national level, school-led projects, assuring resources and grants for graduating students to attend the first three years of undergraduate education. The same students during the three years grant period will serve as near mentors for the students attending school. In this way, the undergraduate students will develop their leadership role (Maiorana, 2022) and, instead of performing unrelated work, they can work on domains related to their learning path and contribute, under the school teachers' supervision to a quality learning path for the students seeking high school graduation. We invite further discussion of this, and further ideas related to the theme, in Italian conferences, starting with the Scientix one, calling for a round table among educational

research institutes, e.g. INDIRE, national evaluation institutes, e.g. INVALSI, ministry, and political representatives.

4.3.5. Importance of social science theories in education

According to the 2019 Eurostat report (Mayer, 2018) in 2017 22,4% of the population of the European Union, almost 112 million of people, are at risk of poverty or social exclusion. The situation, according with UNESCO data, has further worsened with the recent crises, i.e., pandemic (Cristaldi, 2021) and war. A quality education for all imperative requires using education to avoid these exclusions. And the educational process must be supported by social science research.

In the realm of higher education (Connolly, 2020) it was recognized that computing is starting to move from the methodologically singular natural/engineering science into the methodological pluralism of the social sciences. The same applies in the realm of secondary education and, in general for school education. These questions have guided our analysis:

- 1) Do theoretical frameworks from other disciplines require social theories and deep contextual knowledge to explain data patterns and social fragmentation embedded in digital platforms?
- 2) Does computing education courses require the adaptation of social science theories?
- 3) Would the addition of multidisciplinary faculty in computing education enrich the intervention, intersection, and impact of social science in computing education?

Approach

A. *Theoretical Frameworks and Social Science Theories*

Freire's Pedagogy of the Oppressed (Freire, 2018a) is a foundation text for critical pedagogy, proposes a tripartite relationship between educator, student, and society. Freire regarded traditional pedagogy as "the banking model of education" in which a student is regarded and treated as an empty vessel to be filled with knowledge, provided by their lecturers, very much like a piggy bank. He argued that pedagogy should treat learners as co-creators of knowledge. Furthermore, Freire argued that teaching is not mere a transmission model, which information is transferred from one mind to another, but rather creating opportunities to access relevant, actionable critical thinking skills that leads to individual empowerment. Thus, the purpose of a course is not the course in and of itself, but rather what individual students achieve. Therefore, the challenge for computing academics is to ensure that learners have a part to play in the evolution of computing curriculum.

To reach this goal, it is imperative to develop at every level of the learning path critical thinking skills. Critical thinking is a fundamental skill that must be achieved in a school setting that will transition into higher education contexts. For school setting the PISA 2018 in focus 113 (Suarez-Alvarez, 2021), stated that 15 years old students, at the end of the compulsory

learning path in almost all countries, shows a continuing increase of internet usage (Peña-López, 2017), (Peña-López, 2019), but “the opportunity to learn digital skills in school is far from universal”. PISA 2018 posed questions on how to decide if an information found on the web is trustable, how to compare different web pages and decide which information is more relevant to the task at hand, and how to detect whether an information is subjective or biased. The Index of Knowledge of Reading Strategies for Assessing the Credibility of Sources (OECD Publishing, 2021) found students from advantaged socio-economic backgrounds perform better than the disadvantaged students regardless of the country of provenance. This coupled with the demonstrated positive effect (McGrew & Byrne, 2020) of classroom intervention to improve learners' critical thinking skills implied that opportunities to learn critical thinking skills must be offered to all students with a particular emphasis on the disadvantaged. The report (Suarez-Alvarez, 2021) concluded that having well informed citizens able to navigate ambiguity, triangulate, and validate viewpoints are essential to preserve democratic values. The application of critical thinking coupled with computational skills should be applied to discriminate between information, misinformation and propaganda, which in recent studies and massive open online courses (MOOC) have debated where a strategy based on lateral reading, click restrain and use of trusted source with curated references has been reported (McGrew & Byrne, 2020), (McGrew, 2020).

A recent free collaborative book (Ko, Beitlers, et al., 2023) built with an ongoing effort around the idea of “Paulo Freire’s Pedagogy of the Oppressed, who viewed the purpose of education not as imparting knowledge, but as awakening students’ critical consciousness to the systems, structures, and ideas that constrain and oppress them. In (Pérez et al., 2018) the study demonstrated how night grade student’s critical thinking and assessment of information reliability could be improved through multiple text comprehension using an analytic framework comprising dimensions of:

- 1) author position (competence)
- 2) author motivation (intention),
- 3) media quality (pre-publication validation).

As Jenkins seminal work (Jenkins, 2009) suggested to obtain a positive governance of digital environments, and at large of society, it is important to close:

- 1) the participation gap allowing all to have the same access to technology
- 2) the transparency gap allowing all to recognize how media shape perceptions of the world and be able to critically evaluate information;
- 3) the ethics challenge changing educational and socialization practices to better prepare young people for their civic active roles as media makers and community participants.

Because critical thinking is recognized as one key element to navigate the modern “liquid society” (Bauman & Leoncini, 2018) where everything is constantly and rapidly changing, these competencies must be developed for data and media literacy.

The infusion of copious data in daily life dictates an urgent need to foster a culture of data analysis for all citizens. One example of fostering this culture is the “OpenCoesione” Project (Ciociola & Reggi, 2015). This Italian open government initiative utilizes open data to engage students, teachers, and civic society to monitor how public funds are used. The importance of data in education and learning is supported in (Coughlan, 2020) and (Krishnamurthi & Fisler, 2020) research. The centrality of data was emphasized and envisioned with data as a means for upcoming generations of students to communicate across disciplines in the language of computation. Critical thinking supported by data, grounded in theories with a solid contextual knowledge can be the foundation of the new active digital citizens.

To achieve this, the challenge for educators, the learning community, and society is to produce quality content and understand that content is the most crucial resource on the web. Another crucial resource is time to demonstrate and wisely use time to search, read, select, summarize, communicate, disseminate, and produce quality content critically evaluated to have a positive impact on the individual, and society.

B. Adapting Social Science Theories in Computing Education

Advancements in and adoption of data science, artificial intelligence, machine learning, and deep learning have benefited society because the computing programs offered discrete social science modules in those computing topics. However, research and the press have reported the negative impact of usage of computing technologies. For instance, bias in recruitment and selection of potential employees, incorrect facial recognition of individuals, allocation of grades to students, and rejection of mortgage applications. All these examples are a consequence of algorithmic bias. Concerns regarding algorithmic bias in the design of data models, and how the associated collected data is used, have resulted in policy makers and governmental agencies investigating how ethics can be applied to these computing topics. Thus, computing curriculum designers need to consider how to incorporate ethics to address algorithmic bias within their computing curriculum to maintain academic currency, and to ensure that learners are aware of algorithmic bias and how to address it in the design, development, and deployment of any digital artifacts they produce.

A recent work (Jobin et al., 2019) conducted a content analysis of 82 studies and identified 11 ethical principles to incorporate into curricula. The eleven ethical principles are transparency, justice and fairness, non-maleficence, responsibility, privacy, beneficence, freedom and autonomy, trust, sustainability, dignity, and solidarity. The first five, transparency, justice and fairness, non-maleficence, responsibility, and privacy, were referenced in most guidelines identified by the research. Therefore, education could start incorporating these ethical principles based on importance and priority found in the research. Because computing transcends all industries and aspects of daily life, the computing and education community should facilitate interaction between different disciplines to (1) demonstrate the benefit of data and computing capabilities from the scientific community, (2)

explore scientific methods and social theories from the social science community to support the process of data interpretation, and the integration data empiricism.

Computing curriculum designers have been challenged to address environmental sustainability in the United Nation's Sustainable Development Goals (Vincenti & Pecher, 2020). (Kumar & Buyya, 2012) challenge computing educators to define how environmental sustainability can be applied to cloud computing. While (Poongodi et al., 2020) advocate for environmental sustainability in the deployment of Internet of Things (IoT) devices. The European Union recently published a handbook for evaluating the impact of Nature-based solutions⁶⁷ (NBS) (Wendling & Dumitru, 2021), which includes indicators of NBS performance and impact on 12 societal challenges:

1. Climate Resilience
2. Water Management
3. Natural and Climate Hazards
4. Green Space Management
5. Biodiversity Enhancement
6. Air Quality
7. Place Regeneration
8. Knowledge and Social Capacity Building for Sustainable Urban
9. Transformation
10. Participatory Planning and Governance
11. Social Justice and Social Cohesion
12. Health and Wellbeing
13. New Economic Opportunities and Green

JobsSocial cohesion, according to (Uzzell et al., 2002) has been proved to represent an important resource for long-term environmental sustainability. Consequently, socially cohesive communities tend to be more supportive of environmentally sustainable attitudes and behaviors compared with those communities that lack social cohesiveness. These studies support the mandate for student exposure to socially responsible behavior by presenting both the environmental footprint of Computing (Pollock et al., 2019) and Information Communication Technologies (ICT) (Dri et al., 2015) and the positive impact that these technologies can do to mitigate and improve the adverse human effect on nature.

C. Multidisciplinary Faculty in Computing Education

Harmse and Wadee (Harmse & Wadee, 2019) cautioned computing curriculum designers and Heads of Computing about the identification and articulation concerning the decolonization and development of an inclusive computing curriculum. The direction

⁶⁷ https://ec.europa.eu/info/research-and-innovation/research-area/environment/nature-based-solutions_en

indicators must be addressed specifically, as suggested in (Wendling & Dumitru, 2021), on how to enhance “trust, solidarity, tolerance, and respect” which are generally understood as manifestations of a cohesive society, one that works towards the well-being of all the members, that is, towards the common good. The above indicators must be grounded in theories, scientifically tested, and applied to the environment and human beings. The 2016 International Civic and Citizenship Education Study (ICCS) study (Schulz et al., 2018) for 8th grade students found the more students are exposed to civic and citizenship education the better students are able to cope with the recent challenges and the aforementioned issues related to climate change, environmental protection, respect of others in all aspects from religion to politic, and openness to diverse culture and migrants. This study will be continued in the ICCS 2022, which will focus on issues in relation with the United Nations Sustainable Development SDG (SDG) related to education in cooperation with UNESCO participating countries.

Experience reports

This section will utilize a case study on the development and delivery of undergraduate computing curricula designed with social science theories in universities located in the United Kingdom and Italy. Section IV will delve into reflections of the decolonization of the curriculum.

A. United Kingdom

The UK undergraduate computing curricula are designed to uphold validation regulations of the awarding university and be in accordance with the national subject benchmark statements for computing undergraduate programs (QAA 2019) and the professional subject association accreditation (Bowers & Howson, 2019). The University’s validation regulations require program specification documentation to indicate how the program will address bias within the subject, environmental sustainability, and decolonization of the curriculum. In addition, current and former students are invited to have their voice heard and be actively involved in the design of the curriculum as curriculum co-creators. Thus, embracing and adopting the principles of Freire’s Pedagogy of the Oppressed (Freire, 2018b) to have learners influence the design of the curricula. To adhere to national computing benchmark statements, the program team must clearly indicate how unconscious bias within the computing, for instance algorithmic bias, is to be addressed.

B. Italy: students majoring in computing

In Italy extensive, multi-year, multiclass teaching experiences were conducted in the last triennium of high school with students majoring in computer science. During these experiences one of the authors followed the students in their learning experience for their last triennium, directing the team of class teachers and accompanying the students in their final national graduation exams. This allowed to build solid interpersonal relations with the

students, relations endured after graduation during the undergraduate and professional work of some students. These interpersonal relations greatly helped both the teachers and student's well-being at school. It is reported in literature (Borgonovi, 2015) that teachers-student relations are strongly associated with both performance in mathematics and student happiness and sense of belonging to school. Literature exists confirming the same results in college settings where social support is reported to be one protective factor for test anxiety (Hyseni Duraku & Hoxha, 2018).

The educational experiences in grades 10-16 were characterized by:

1. A high degree of innovation in the content, updated every year, covering aspects related to technical and scientific topics supported by selected reference papers found in scholarly work. This applied to initial programming, object-oriented design, and database and web programming classes. Innovations in learning trajectories are suggested for initial programming in (Maiorana, 2021c), (Maiorana, 2019a) for object oriented programming (Giordano & Maiorana, 2013e), (Giordano & Maiorana, 2014e) and for database and web development (Maiorana, 2014), (Maiorana & Giordano, 2013b), (Giordano & Maiorana, 2013f).
2. By a variety of pedagogical approaches ranging from inquiry-based learning, flipped classroom, problem, project and challenge-based learning, learning by doing and experiential learning, peer instruction, process oriented guided inquiry learning, pair programming, and in general active students pedagogies. The beneficial effect of the blending of student-centered active pedagogies was also reported in a recent work related to an online teaching experience in the USA (Maiorana, 2020).
3. By a rich set of technologies including a different class site, used in every class from 2012. The class site was used by the students to construct their portfolio of activities. The site allowed centralizing in each page, lesson discussion and problem-solving lab activities, with questions and answers resolved by the instructors and peers. The homework and lab activities were also formally assessed and commented on by the instructors and, with the permission of the students, the suggestions were visible to all students. The Google analytics capabilities gave the possibility to track student usage to spot early signs of disengagement, usage persisted and tracked even years after the class ended. Vast was the range of course specific technologies ranging from XNA game development platform (Giordano & Maiorana, 2013e), query plan in Microsoft SQL Management Studio (Giordano & Maiorana, n.d.), flowchart tools, such as flowgorithm (Cook, 2015), and logging mechanism, such as black box extension in BlueJ (Giordano & Maiorana, 2015a).
4. By a rich set of activities and student chosen real life projects involving student chosen open real datasets such as MusicBrainz (Swartz, 2002) (Poongodi et al., 2020).

In an undergraduate context in Italy a three-year experience on a first course in Computing for humanistic students, a shared pool of projects was created where students shared their projects mandatory for passing the exams. The key innovations in this course were:

- 1) The freedom for students to choose their project according to their interests, social and professional expectation. Most of the projects were chosen with a social aspiration, embedded in the local territory and some of them resulted in the design of entrepreneurial applications. A same pattern was observed in a master course for engineering Student (Giordano & Maiorana, 2014a), (Wendling & Dumitru, 2021)
- 2) The same freedom was left to the student to choose the topic of their review paper, also mandatory to pass the final exam.
- 3) The growing of this shared pool of projects during the experience and the increase of projects' complexity.

Finally in Italy the decolonization of the high school curriculum has produced a bylaw threshold of autonomy in designing the curriculum. This, coupled with a by-law tighter integration with industry and learners' participation in on the field work experiences has produced beneficial effects when a careful design of activities has been put in place.

Guidelines

Computing education practitioners, who are considering the integration of social science theories in the computing curricula, the following student-centered approaches are recommended:

- 1) Adopt a problem, project or challenge-based approach favoring an evidence-based assessment
- 2) According to Freire's Pedagogy of the Oppressed, decolonizing the curriculum by letting the student play an active role in choosing the project topic, find partnership in the local community or in broader context, privilege stages and activities favoring entrepreneurial initiatives. It has been discussed above that students, especially the youngest, tend to favor projects for social good. The freedom to choose the project allows students to easily take ownership and accountability⁶⁸, i.e. "acknowledgment and assumption of responsibility for actions, products, decisions, and policies of the work and increase intrinsic motivation because "they are fully engaged in a task for reasons inherent to the work itself"(O.E.C.D., 2021). According to (O.E.C.D., 2021) research has demonstrated the positive impact of intrinsic motivation to creativity and to the educational learning process.
- 3) Build a shared pool of peer and instructor assessed projects. This facilitates students' collaboration and communication and near mentorship practices: students attending

⁶⁸ <https://en.wikipedia.org/wiki/Accountability>

a successive edition of a course, once they choose a project, naturally tend to communicate with the developers of the project. With the shared pool of projects students' progress.

in their learning journey are visible both during the class time and longitudinally.

- 4) Be inclusive. According to the Humanitarian Free Open-Source Project guidelines (H. Ellis et al., 2008), (H. J. Ellis et al., 2015a) every project should require many competencies thus allowing a concrete contribution by students not majoring in computing and, at the same time, be exposed to core computing principles. Including theories from the social and humanistic sector will allow every learner to apport a unique contribution.
- 5) Let the project be interdisciplinary and multidisciplinary in nature by favoring collaboration between different teachers, instructors, and educators in the same or different institutions. This is natural in a school setting, like in Italy and in some European countries, where students attend a class with a pool of teachers shared by all the class students which all attend the same lessons during the whole school time. In contexts like in the USA and in an Anglo-Saxon setting, where each student has a complete freedom to choose each class attended in a school day, it is a bit more difficult but it can be favored by teachers and educational managers. Student led initiatives for intra and interclass collaboration should be supported.
- 6) Encourage collaboration and socialization. Let the students freely choose the group composition and then favor the integration of groups by merging projects and by sharing experiences among two or more groups requiring students to work for a period inside other groups.
- 7) Integrate flexible formal assessments. Give the student several possibilities and time slots to present their work for peer and instructor assessment. In class assessment activities greatly favor student critical and creative thinking. Support students' assessment by a clear methodology and a flexible rubric. Instruct them to give great and constructive feedback. In this way extrinsic task motivation and deadlines pressures are reduced. According to (O.E.C.D., 2021) research has demonstrated the negative impact of extrinsic motivation to creativity and to the educational learning process.
- 8) Adopt an agile project development and assessment. Favor frequents in class student presentation of their work with a pitch format, from 1 to 5 minutes presentation) for formative assessment, up to longer formal assessment activities. Include formative and summative peer assessments.

The curriculum design must foster critical thinking in learning text and data, which includes all forms of multimedia and transmedia communication. Hence broadening the computing curricula with aspects related to social science and humanistic aspects will enhance reading and math competencies required to develop critical citizens involved in civic active

participation for the social good. The educational process must support the development of all four paradigms of science (Kitchin, 2014), i.e. experimental, theoretical, computational and exploratory combining data supported empiricism with grounded theories pioneered by sociologist, anthropologist, psychologist by prominent educators from the classical age to the modern and contemporary age.

A critical tool for combining the educational process while engaging students is dramatization. For example, students are asked to interpret the roles of characters in courses that study literary works. This approach has an extraordinary value because students learn how to evaluate another person's position. This type of classroom exercise allows the shiest student to submerge themselves into this experience with an exciting strength and vitality, which overcomes many barriers, such as distrust of others and reluctance to get involved. Taking a different perspective from one's own, even if only for the duration of a lesson, is important because it opens, once again, not only to welcome those who are different from us but also to find new spaces for building a relationship on both sides.

Project-based learning is used by computing educators during multimedia and transmedia artifact assignment and for developing and sharpening soft skills competencies (Woodward et al., 2010). This real-world work environment assigns students to different roles that are involved to complete the artifact. Educators find there is an interesting phase based on how students respond to the same task but develop a collaborative original approach that follows the characteristics and skills of the members of the group. Therefore, the final product illuminates the group collaboration through the unification of the music, sounds, video, images, dialogues, setting and the variations on the theme or poetic licenses. Upon completion of the artifacts, a lesson is dedicated to the vision and sharing of the materials produced by the students: it becomes a day where everyone observes and comments on the work of the various groups developing critical thinking. Educators have witnessed this type of collaboration reinforces self-confidence, enthusiasm, sharing of compliments, active listening, enhanced decision-making strategies, and troubleshooting techniques, which are the soft skills employers are seeking (Woodward, Sendall, & Ceccucci, 2010). It is extraordinary what students accomplish when they manage their work autonomously and creatively and is an example of the decolonization of the curriculum according to Freire's "pedagogy of the oppressed", where the students have a privileged role in the creation of knowledge.

Table 1: Overview of social science theories with suggestions regarding the pedagogical, techniques, computing education topic, and instructional example

Social Science Theory	SST Overview	Pedagogical Techniques	Computing Education Topic	Instructional Example
Action Theory	Individuals are active, complex, and react to the social structures around them.	All techniques could be applied depending on course level and students' skill set. See Table 1.2	AI, HCI, Robotics	Students will design, build, and/or present how the development of an AI, HCI, or Robotics project would represent or aid in the response and/or protection against a public health threat.
Systems theory/Functionalism	Study of society as a complex arrangement of elements, including individuals and their beliefs, as they relate to the whole		Algorithms, Database design and collection	Assign teams to develop database designs to collect and analyze demographic information or review a current database; ex. Alabama Tornado Database, Social Vulnerabilities Index, or National Crimes Index, to determine if relevant data is being collected, and if changes are needed.
Psychoanalytic Theory	Explores the unconscious mind to relieve painful emotional symptoms and increase self-awareness, which is the clinical application of Freud's theory.		Multi-media artifact and presentation	Students are asked to develop a multi-media artifact or presentation. After the presentation, discussions on how their use of images and music impacted the audience and what was effective and what was not and why.
Symbolic Interactionism	View of social behavior that emphasizes linguistic or gestural communication and its subjective understanding, especially the role of language in the formation of the child as a social being.		Gamification application focused on learning algorithms, coding.	Students will design a game to teach algorithms or coding that will transcend languages or disabilities.
Rational Choice Theory	States that individuals use rational calculations to make rational choices and achieve outcomes that are aligned with their own personal objectives.		Project-based learning artifact involving data collection and analysis	Student teams will be assigned a data dump that is relevant to a real-world scenario and are asked to arrange the data and create the algorithms for analysis to provide scientific and reliable outcomes.

Phenomenology	Within sociology, Phenomenological sociology) is the study of formal structures of concrete social existence as made available in and through the analytical description of acts of intentional consciousness.		Client Data Specifications	Students will review a case study and design questions they feel are important to successfully design an application, database, or analysis their client is requesting from their team. After the collection of specifications, the team will create a prototype of the application or database and send it to their "client" for feedback to determine how well they understood their clients' needs.
---------------	--	--	----------------------------	--

4.4 A second computing course on algorithms and data structures

In this work, by leveraging on previous experiences (Maiorana, 2019a), (Giordano, Maiorana, Csizmadia, Marsden, Riedesel, & Mishra, 2015a), (Giordano & Maiorana, 2014d), (Giordano & Maiorana, 2015a), (Giordano & Maiorana, 2013e) we will consider and address the following challenges faced in designing and developing a fully online course for undergraduate students not majoring in Computer Science:

- 1) How to deliver a fully online course
- 2) How to design the content so it is suitable for students not majoring in computing
Ideas and thoughts will be shared on:
- 3) How to integrate different approaches like unplugged activities, puzzle-based, and coding activities into Algorithms and Data-Structures (A&DS). How this variety of design and implementation tools can serve learners with different learning styles.
- 4) What activities, tools, and pedagogies are best suited for a formative assessment approach to A&DS education.
- 5) How A&DS can be used to teach core CS ideas and 21st-century skills with an interdisciplinary approach that can serve students, teachers, and researchers with different backgrounds, interests, expertise, and experiences.
- 6) How to use A&DS to expose students to different areas with new and quickly changing domains

4.4.1 How to address online learning

The main strategy used in designing the course was a focus on supporting student-content interaction. The content was provided by means of an online book supported by an Open Online Course with video lessons, assessments, and coding activities.

Student-content interaction is supported by means of:

- 1) Animations and assessment activities supporting student self-reflection
- 2) A flipped approach to the content development: start from the formative assessment, proceed with the content and assessment solutions

Online learning in this time of crisis due to the presence of COVID-19 has become a focus of learning communities. The course is the third one in the program and, according to modern Human Computer Interaction design principles, the choice was made to use a consistent look and feel across the whole program. Regarding online learning, the following suggestions and lessons learned are shared:

1. The course was designed for an expected audience of more than 50 students. For this reason, the decision was made to use asynchronous delivery of lectures. In this respect, asynchronous lectures give more freedom to students to follow the course at their own pace. To mitigate the effect of students' procrastination, it is advisable to decrease the course load at end of the term and give tighter deadlines at the beginning, releasing this deadline at the end. This coupled with other interventions (J. Martin et al., 2015) can contribute to students' success in projects.
2. The course was designed to be fully online. Office hours were held, using synchronous Zoom sessions. Teaching assistants were available, according to students' requests, for face-to-face meetings. Our suggestion is to offer, whenever possible, face-to-face session with students coupled with on-campus day-long activities to be attended by students on a voluntary basis.
3. The assessment in all its forms, from formative to summative, covers an important part of the course. The formative assessment was designed with open and closed responses. The summative assessment was project-based with the grading of students' projects performed using an auto-grader to ensure scalability. Coupling the auto-grader with human review allows for better learning experiences, allowing students to benefit from the feedback. Organizing synchronous online sessions for students' projects presentations in front of all classmates, divided in groups where necessary, offers an invaluable means for students to improve their communications skills, receive feedback from peers, and from the instructors.

4.4.2 How to address students not majoring in Computing

The importance of involving non-majors in computing education has been exposed in (Guzdial & Forte, 2005). The approach used to introduce computing to non-majors has been explored for several decades. In (Guzdial, 2015), (Han et al., 2019), (L. Porter et al., 2018) several approaches based on engaging practices and by considering different populations of learners addressing their learning goals and exposing them to the communities of practices they are more interested in have been presented. The approach taken for the course followed by non-major students was to:

1. Offer the core algorithm and data structure content in a rigorous way
2. Add diversity offering a rich set of activities and projects, such as application of the algorithms and data structures in different domains, adding applications in a modular way. The diversity must span both the application domain and the technologies used, offering a first insight into advanced topics such as parallel programming in the simplest and most accurate way possible. Allowing for students' contributions and project proposals is a suggested approach for their involvement.
3. Use of data and real-life datasets in different domains with an insight into the techniques used to analyze data in different domains: text and humanities, numbers in Science, Technology, Engineering and Mathematics (STEM), audio, images and multimedia as engaging activities.

4.4.3 Course content

The course on Algorithm and Data Structure was designed and developed inside the Computational Core certificate⁶⁹ at the Kansas State University as the first course of two focusing on the topic. It is the third course in the program after two courses on covering an introduction to programming, the first one, and a second course on algorithm and programming. In designing the content, the priority was given to favoring student-content interaction. This interaction must be coupled with peer interaction and student instructor interaction.

The content of this course must cover basic linear data structures leaving more advanced non-linear data structures to the successive course, the second on data structures and algorithms, the fourth in the program. A fifth mandatory course should cover software design principles with a capstone project. Elective courses can be selected by the students among a set of available ones, covering databases, data science and other topics. The author was not directly involved in the design of the whole program and details on this will be shared in a successive work.

The basic concepts of object design and programming are presented, after reviewing, and consolidating the key concepts of imperative and procedural programming, from the very first lesson through examples and simple applications. The concepts presented are applied through the book guiding students, with an incremental approach, in realizing software projects with increasing difficulty and complexity that require the use of multiple classes in relation to each other and one or more objects of each class. According to modern software development approaches, the incremental approach allows for the building of a working application at the end of the first module which will be further developed along with several modules, adding functionalities, and refactoring i.e. modifying the code structure without modifying the program behavior in order to make it easier to read and extend.

⁶⁹ 1 <https://core.cs.ksu.edu/authors/>

The presentation of data structures is done both by introducing the properties and the operations allowed by each data structure, in order to make the reader able to implement the data structure, and through the use of the developed data structures by applying them in projects requiring their use. Some projects will also require a starting use of library and Application Programming Interfaces. Algorithm design techniques have been presented through puzzle-based activities (Levitin & Levitin, 2011). Collaborations with other initiatives were actively pursued to sustain the effort in developing the book. Elective activities have been presented by proposing projects with an interdisciplinary context. Other activities will be added in further developments of the course for a successive use in further classes.

4.4.4 Pedagogies

The main pedagogical approaches adopted are:

1. Active learning and inquiry-based approach
2. Incremental learning at all stages from design to project development and assessment
3. Use of animation to scaffold students' self-reflection ,
4. Facilitate students' engagement though offering a broad range of activities in different domains

The pedagogical approach is based, according to modern active pedagogies, on an inquiry-based approach. Leveraging on the author's experience (Maiorana, 2019a), (Maiorana, Csizmadia, G., et al., 2020) students' engagement has been pursued through formative assessment activities to be completed before content presentations to implement a flipped book centered on a cycle of activities starting from student self-reflection.

The incremental approach in the formative assessment has been pursued to scaffold student self-reflection on new concepts, guiding them through a deeper level of insight and technical detail. The formative assessment activities have been scaffolded by animations extensively used to display algorithms and data structures operations. The approach used was to let the students watch the animations and then answer the assessment activities.

The same incremental approach has been used in project development supported by design, code, test, debug, and document activities. Examples in this direction are:

1. a scientific calculator that started from the basic arithmetic operations implemented using assignment to zero, increment and decrement by one, coupled with the control instructions (conditions and loops). The project was then refactored using functions, adding other operations ranging from power to factorization and primality, designed in an object-oriented fashion, implemented recursively and extending some operations like division with decimal using non-linear data structures such as Finite State Machine (FSA).
2. Object-oriented design (OOD) and Object-Oriented Programming methodologies (Booch, 1991) applied to the development of a management of a collection of objects. An example in this direction is reported in figure 3.

3. Use of recursion as early as possible and use of recursive approaches to implement operations of the studied data structures as well as use of stack to understand implementations details of function calls and recursive programs.

4.4.5 Technologies

The technologies used were chosen to support the design choices and the pedagogical approaches:

- For online content delivery: Canvas (Wilcox et al., 2016) for its support of classroom activities and learning management system capabilities
- For online programming Environment Codio: for allowing a programming environment accessible from any device without requiring system level abilities to install development environments and tools.
- For online project assessment: an auto-grader, constructed in-house
- For design of algorithms: Flowcharts (Cook, 2015) for their versatility supporting automatic translation into many programming languages among which the two course chosen languages: Java and Python. Flowcharts are also used in Object Oriented Programming as a way to design method. Raptor has some functionalities for this (Carlisle, 2009).
- For Object-Oriented Design (OOD) the Unified Modelling Language (UML) software the reader can use ArgoUML a free open source software suggested by Free Open Source Software or StarUML (Wong, 2007).
- For creating animations: Edgy coupled with Java and Python Turtle libraries used to extend each data structure class with drawing functionalities used to animate its operations.

4.4.6 Outcomes

The content has been developed in 12 modules suitable for a course on algorithms and data structures covering the major linear data structures, namely stacks, queues, lists, hash tables and the applications of these data structures to implement sets, dictionaries and finite state machines. Recursion is presented at the beginning of the course in order to allow students to familiarize themselves with this difficult concept. In order to build a bridge toward the planned successive course on hierarchical data structures, the discussion of Finite State Machine and their implementation with adjacency matrix and adjacency list have been used to anticipate and introduce the representation of graphs and trees. The basis for elective activities has been laid for further development in the successive course. Figure 1 summarizes the planned content.

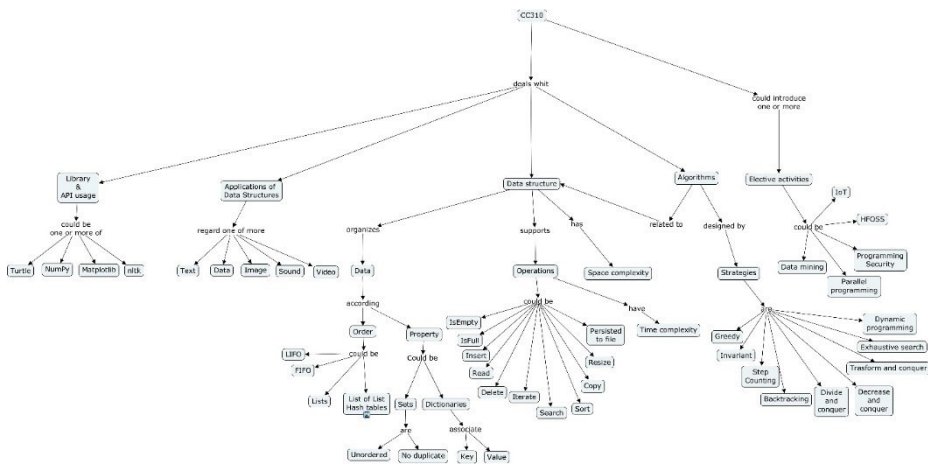


Fig. 1. The course contents

The content has been used for a first edition of the course, named Computational Core 310, at the Kansas State University and is available online⁷⁰.

For describing data structures, the same approach was used. Students were exposed to the underlying properties and operations allowed in each data structure. Figure 2 shows the concept map summarizing the approach used for data structures.

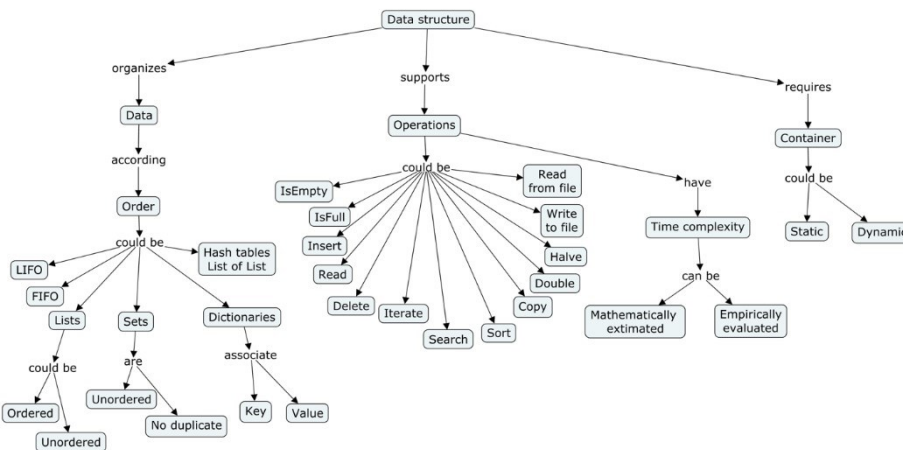


Fig. 2. Couse Data structures overview

To facilitate the use in different contexts, the libraries available in the two main programming languages used in the course, namely Java and Python, were presented and applied to solve problems in different domains. Figure 3 shows the approach used to present and apply language specific data structure libraries for the Python language.

⁷⁰ <https://core.cs.ksu.edu/3-cc310/>

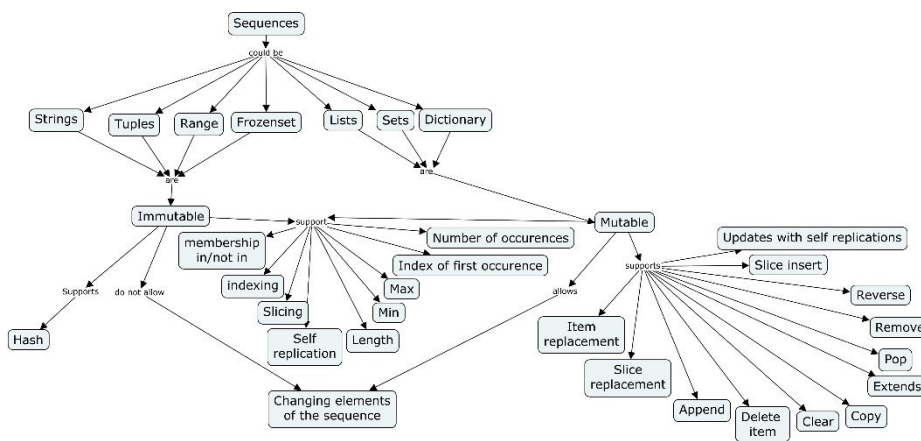


Fig. 3. Python libraries overview

A similar approach was used for Java libraries. For presenting the object-oriented approach, a methodology presented in (Booch, 1991) was used coupled with an incremental approach: as a recap, a class, with attributes and methods, presenting issues related to information hiding, was introduced. After this, inheritance was presented by adding to the project subclasses. The relation with information hiding as well as multiple inheritance and related issues were addressed. The following steps consisted in presenting hierarchies such as composition, avoiding, at a first stage, to point out advanced distinctions with aggregation, followed by use relationships and associations. Figure 4 shows an example of a completed project proposed to students with the above described incremental approach.

The use of activity diagrams could be suggested as an elective activity and to assist students facing difficulties in following the flow of execution and object interactions. The whole course syllabus is reported in Table 1 with sample activities and project suggested as mandatory or as elective. According to the design principle above mentioned, reading material such as (Goldweber et al., 2019) should be suggested to the students as well as many resources as possible in order to give them the ability to choose the projects they like the most and contribute with their projects to forming a shared memory of resources developed by the students inside the class.

The course content is summarized in Table 1 where the tools and suggested activities, either mandatory, or elective, are detailed.

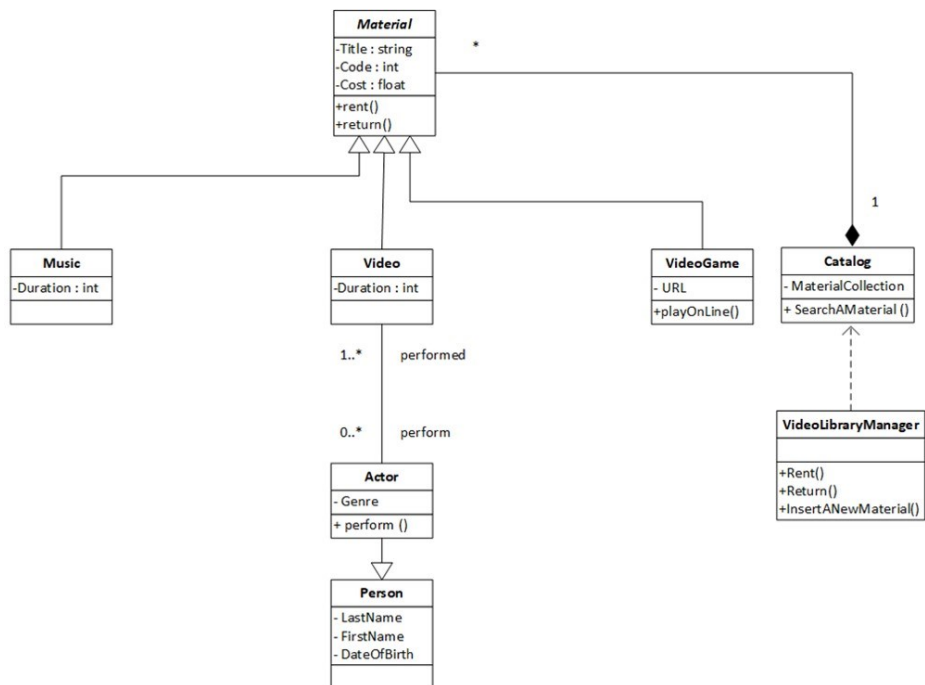


Fig. 4. An example of Object-Oriented project

Table 1. Course content, tools, and activities for each module

Mod.	Content	Tools	Activity
1	Review of imperative programming	Flowgorithm (Cook, 2015)	Computing calculator (Calc) with assignment to zero, increment and decrement
2	Review of Object-oriented programming	Flowgorithm, Raptor (Carlisle, 2009), StarUML (Wong, 2007)	Refactor computing calculator using OOD. Implement the project in Fig. 3
3	Programming by Contract & Introduction to Performance	Python and Java libraries supporting programming by contract, e.g. ⁷¹	Find pre, post and invariant conditions of algorithms, e.g. the computing calculator
4	Data structures and Algorithms	Edgy	Implement a brute-force technique to solve a strategy game
5	Stacks	Edgy. Java and Python turtle libraries	Extend Calc to handle expression.
6	Recursion	Flowgorithm, Animations, CargoBot (Tessler et al., 2013), (Lee et al., 2014)	Refactor the recursive calculator.
7	Searching and Sorting	Flowgorithm, Animations	Sorting and searching applications
8	Queues	Animations, Edgy	Priority queues management
9	Lists	Animations, Edgy	Set and set operations on real-life datasets
10	Hash table and Dictionary: properties and implementations	Animations, Edgy	Translation service. and WordNet usage Dictionary implementations using data structures
11	Sets: property and implementations	Animations, Edgy, Flowgorithm	Set implementations using data structures
12	Finite State Automata: Property and implementation. Data structure comparisons	Animations, Edgy, Flowgorithm	Extend the CC calculator to handle decimal divisions. Design and implement an augmented data structure

Giving a recap and a broad overview on data structures at the beginning of the course allows for easier acquaintance with the new material. Having a broad overview allows for a more proactive student role. The design process should be sustained at all levels coupled by offering different possibilities to approach the content from puzzle-based activities to the use of block-based languages for fast prototyping. Regarding the learning trajectory, anticipating recursion at the beginning of the curriculum had the benefit of covering an important topic

⁷¹ Python programming by contract libraries

earlier, allowing more time for students' reflections, sustained by continuous use of a recursive approach in the course module, and avoiding student cognitive overload.

Use of recursion, complexity analysis and invariant identification should be applied transversally along all the modules of interest. According to (Maiorana, Csizmadia, G., et al., 2020), these analyses should help the students reach the learning goal of selecting and comparing different data structures and analyzing the complexity of the algorithms used.

A second work focuses on comparing recursion with iteration as they are perceived by learners in a first computing course. It also attempts to identify when is the best time to teach recursion and compare both iterative and recursive design techniques. As a case study, the authors utilised an attitude survey to be completed by the participants to determine their perception of both recursion and iteration implementation for both Bubble and Merge sorts, and a module which focused on recursion compared with iteration. In addition, we report on the design and development of animations for both the Merge and Bubble sort algorithms to illustrate and illuminate these algorithms. Three different formative assessment tools are introduced that can be used to sustain and guide students throughout a self-explanation of the two algorithms. A first run of the formative assessment tools have been administered to students in a first computing course: the first with 15/16-year-old high school students and the second with undergraduate students majoring in humanistic studies in their final year before graduation. By using both animations and an inquiry based approach the work aims at investigating the most appropriate time and way to teach sorting algorithms using both recursion and iterative algorithms.

4.4.7 Attitude survey

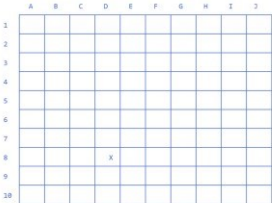
There exist computing attitude surveys that span the learning cycle from middle school (Csizmadia et al., 2019c), (Maiorana et al., 2015a), (Román-González et al., 2018) through to higher education. We have developed an attitude survey that, according to McCauley et al. (McCauley et al., 2015) tests the students' attitude, first to evaluate and trace recursion, and then to write recursive functions that reproduce a given pattern. The attitude survey questions are programming language independent with all the questions, including those related to tracing, are expressed in natural language. Some examples of the tracing questions used are reported in Table 1 and Table 2.

The questions can be used as a pretest before the module on recursion and how it relates to iteration is delivered and as post-test at the end of the same module and at the end of the course itself. If used at the beginning of the studies, for example at the beginning of the undergraduate studies, then it should be used in conjunction with an attitude survey (S. Fincher et al., 2006) such as Computing Programming Aptitude Test developed by the University of Kent that focuses on logical thinking and problem solving, pattern and syntax

recognition, and ability to follow complex procedures. The questions proposed in Table 1 and Table 2 belong to the last category.

The attitude survey and the following discussion allow us to present different ways to repeat actions within programs, from using loops which allow code to be iterated a fixed number of times, when either a condition is met, or until a condition is met, or recursive functions that call themselves until a base case is reached. With a similar approach it is possible to ask the students to reproduce a drawing by designing either a recursive or an iterative program. Such type of activities has been effectively implemented within a blended learning approach to extend the educational activities beyond the time and space class limit.

Table 1. Some examples of tracing recursive function questions

<p>In the following board if the x represents the Global Positioning System (GPS) location of JoCS, how many different cells will JoCS touch? Will JoCS stay inside the grid after executing the movements? List the different squares touched by JoCS in order of time after executing the movements in a). If a square is touched more than one time, list it every time it is touched. List only the squares inside the grid. Separate the squares by a comma without spaces. In writing the square list the letter then the number, e.g. A1.</p>		
		
<p>a. 1 step to the right</p> <p>b. Call Help (3) Where Help (N) is: If $N > 0$ then Go one square left Go one square down Call Help (N -1)</p>	<p>Call Help2 (4) Where Help2 (N) executes If $N > 0$ then Call Help2 (N -1) Go 1 square down Go 1 square right Go 1 square down</p>	<p>Call Help3 (3) Where Help3 (N) executes If $N > 0$ then Go one square right Call Help3 (N -1) Go one square down Call Help3 (N -2)</p>
a) Tail recursion	b) Head recursion	c) Tree recursion

4.5 The recursive module: content, technologies, and learning trajectories.

We designed a module for teaching recursion module that was inspired by the following design principles:

- Using an inquiry-based approach to design the module with a flipped learning approach, which started with a scaffolding formative test soliciting student reflection on key concepts. This test was used as a pre-test for this investigation. Example of activities that can be used with an inquiry-based approach are:
 - The attitude survey, which is presented in Section 3.
 - Animations, which are discussed in Section 6 used along with formative test guiding students in correct descriptions of what they have seen in the animation. Ensure that the content is as accessible as possible in order to use at the beginning of the course. This provides students with an opportunity for a longer self-reflection period and provides them with multiple possibilities to practice, developing in parallel both the recursive and the iterative

solution to the problem they are presented with. These are examples of the positive effect of introducing recursion and there are others in literature such as (Gunion et al., 2009).

Table 2. Examples of tracing iterative algorithms: a) repeating a fixed number of times; b) repeat until condition, and c) use function with repetitions.

<p>In the following board (similar to the one used in Table 1) assuming that the x represents the Global Positioning System (GPS) of JoCS, in which cell JoCS arrives after executing the following movements?</p>	
<p>a. Repeat 4 times a. 1 step to the right b. 1 step pointing up</p>	<p>a. Until not in the last column a. 1 step to the right b. 1 step pointing up</p>
<p>a. 1 step to the right b. Call Help (4) Where Help (4) executes a. Repeat N times a. 1 step to the right b. 1 step pointing up Assume that N has the same value of the value used to call the procedure Help (4 in the example)</p>	

The taught module has been experimented with both visual block-based programming languages, for example App Inventor, as reported in the case study, and with textual programming languages, for example Python and Java, within an online course on data structures and algorithms for non-major (Maiorana, 2020) students.

4.5.1 The content

We designed and developed a teaching module to be developed as part of a data structures and algorithms course. The authors recommend a two-week schedule for the whole content.

The content of this module is outlined below:

1. Introduction: recap on iteration:
 - a. Repeat a fixed number of times
 - b. Loops and conditions at the head or at the tail
 - c. Repeat while a condition is met
 - d. Repeat until a condition is met
 - e. Repeat for all the elements of a list
2. What is recursion?
3. Some examples:
 - a. draw a square and generalize it to a regular polygon
 - b. invert a sequence of characters
4. Implementing recursion:
 - a. LIFO function call order

5. The structure of a recursive program:
 - a. The factorial
 - b. recursive version of the algorithm to find the max and the min of N number
 - c. recursive merge of two ordered lists.
 - d. Project: a recursive calculator using only assignment to zero, increment and decrement
6. When it is not worth using recursion:
 - a. the Fibonacci numbers
 - b. naïve implementation
7. Solving recurrence equations
 - a. Time complexity of the Fibonacci number:
8. How to improve recursion: memoization
 - a. Fibonacci performance improvements.
 - b. Retake of the recursive calculator with performance improvements
9. When recursion is worth using
 - a. The tower of Hanoi
10. Transforming a recursive program into an iterative program

4.5.2 Case study

Strategies for teaching and learning computing have been considered by the authors from many different points of view (Luxton-Reilly et al., 2018) and different learning trajectories have been explored for a first computer science course (Rich et al., 2017). One approach investigated relies on animations of algorithms to facilitate the understanding of algorithms (Urquiza-Fuentes & Velázquez-Iturbide, 2009). Animation has been integrated into programming environments for program visualization (Myller et al., 2007). This work describes an experience relating to the design of animations for both bubble sort and merge sort algorithms and the design of formative assessment tools rooted in previous experiences (Giordano, Maiorana, Csizmadia, Marsden, Riedesel, & Mishra, 2015a). (Oates, Coe, Peyton-Jones, et al., 2016a) that an inquiry-based approach (Nature, 2010), (European Commission et al., 2007), (Hamouda et al., 2017), (Hazelkorn, 2015c), (L. Porter et al., 2016a), (B. Simon & Cutts, 2012) can sustain students in their learning of the algorithms and the underlying technique, i.e. iteration and recursion, during a first Computer Science (CS) course. Animations and formative assessment tools will be used to investigate the following two research questions:

1. Is recursion perceived, by students in a first CS course, more difficult to grasp than iteration?
2. Is an active learning approach where students first explain the steps within an algorithm just from observing its animation and then are exposed to a detailed explanation more effective than simply exposing students to the algorithm explanation from the onset?

In order to validate the animations and the formative assessment tools, we collected data from an investigation with one group of 15-16 year old high school students and data from another investigation involving undergraduate students majoring in humanistic studies. Both two groups were engaged in their first computing.

4.5.3 Animation design

To design and implement animations, many approaches can be followed: from using a visual block-based programming language such as Snap! (D. D. Garcia et al., 2012b) or App Inventor (E. Patton et al., 2019), to a text-based language with an object-oriented capability such as Python or Java. A block-based programming language allows students to focus on semantics rather than syntax. Furthermore, according to constructionism (Harel & Papert, 1991), it is possible to ask students to read the source code and use it as a reference point for developing their animations. Since the animations are designed to be used both within a first programming course and within a more advanced course where data structures are presented, then the respective audiences' needs need to be carefully considered at the design stage for each animation. Each animation must be explicit, clear, and precise, and still retain sufficient details necessary to avoid any misconceptions, such as showing the swapping process. Efficiency issues must be avoided for the sake of clarity of each animation.

4.5.4 Design of the formative assessment tool

We designed three formative assessment tools to be used with each algorithm. Through a sequence of open scaffold questions, with a different level of insight, aims at guiding students, using active learning activities (Prince, 2004), to obtain insight into each of the two algorithms. In particular, the three sequences of formative tests are outlined below. The first sequence of tests asks the individual student to describe what the animations do and how they work. Then students rate the perceived level of difficulty and a comparison of the two algorithms in terms of simplicity to grasp, learn and code is also asked. These ratings are asked at the end of each sequence of tests. The second sequence of test presented to each student concerns the recursive algorithms:

1. To identify the number of phases, i.e., recursive division and union of the two ordered halves.
2. To describe in natural language each of them.
3. To aggregate the operations during the merge phase in order to identify repetitions.
4. To count the number of operations in the union phase.

Then individual students are asked about the iterative algorithm in order to facilitate the individuations of nested loops:

1. To identify and describe the swapping process.

2. To identify, describe and count the repeating operations.
3. To identify the halting conditions.

The third sequence of tests presented to individual students asks them for the recursive algorithms; with the focus shifting from the merge phase to that of recursive division:

1. To describe the recursive division process and its backtrack.
2. To write the instructions for the merging phase, using a scaffold approach. For the iterative algorithms, students demonstrate a deeper understanding of the nested loop by writing the instructions for the merging phase. The three tests could be administered in different ways, such as at the beginning of the course to establish a baseline, in the middle to measure instruction, and at the end of the course to evaluate students' progression inside the course, and/or for a final comparison of the two algorithms. To evaluate longitudinal retention, a test could be administered at the beginning of a second computing course, such as a CS2 course. There is also an opportunity to administer a third longitudinal evaluation within a first software development course. This is usually delivered after CS2 where recursion, sorting, searching and all the major recursive data structures have been covered, and could complete the longitudinal study on recursion, searching and sorting.

4.5.5 Preliminary field test

The formative assessment tools presented were used in a first CS course for both: a) final year undergraduate students majoring in humanistic studies; b) 15-16-year-old high school students in the second year of their studies. Each animation was shown twice to each group. At the end of the second view, the associated formative assessment test was administered. The animation for merge sort algorithm was shown first, then the bubble sort animation was shown next. Prior to the participants completing the final comparison test, the two animations were shown in reverse order.

4.5.5.1 Finding for the field test with Undergraduate students

Out of the fifty students (n=50) participating in the study, for the majority (95%) of them it was the first CS course in their learning career. In the test administered at the beginning of the course 50 students, (90% were female) completed the tests for the iterative, the recursive algorithm and the comparison. The merge sort algorithm was perceived as the easiest to grasp (60%), to learn (62%), and to code (62%). Overall, at the beginning of the course, it was preferred by 62% of students. Figure 1 reports the students' ratings of simplicity, using a 10-point scale (1 lowest simplicity, 10 highest simplicity) regarding technical details related to Merge Sort and Bubble Sort.

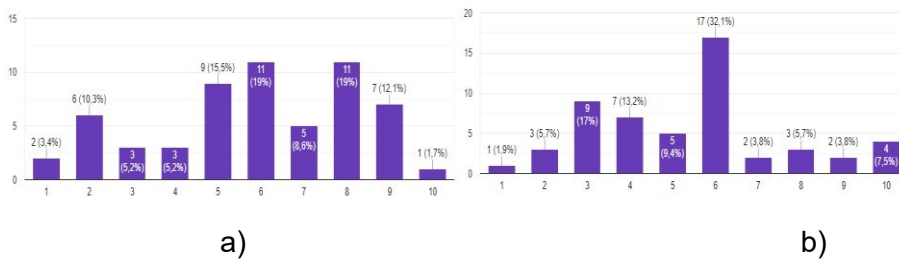


Fig. 1. Perceived simplicity of Merge a) and Bubble sort b) by the undergraduate students

These percentages remained almost steady till the end of the course. A preliminary analysis of the open questions reveals a fair grasp of the concepts underpinning the two algorithms and the necessity for a deeper scaffolding related to complexity issues.

4.5.5.2 Finding for the field test with high school students

The tests were administered to a second-year class of 15-16-year-old high school students studying at a technical high school at the end of their second month of the delivery of that class. For the students, this was their first exposure to sorting algorithms. None of the authors were involved in teaching these students. Ten students out of 20 volunteered to complete the test. The merge sort algorithm was perceived as the easiest to grasp (80%), to learn (70%), and to code (50%). Overall, it was preferred by 70% of the participating students. Figure 2 shows the perceived simplicity of the two algorithms. In this case the Merge sort algorithm was perceived simpler by 80% of the students and it was the preferred algorithm in terms of facility to learn and code.



Fig. 2. Preferred algorithm in terms of facility to learn a) and code b) by the high school student

The content of this section reports experience on designing, developing and field testing of animations for both a bubble sort and a merge sort, and three formative tests to obtain an insight into the perceived difficulties of recursion by students in their first CS courses. The main conclusion that can be drawn from this study is that the students' perceived difficulty of recursion and recursive algorithms is not too high, compared to similar iterative algorithms. Thus, the data sustain the hypothesis that recursion can be introduced as early as possible within a computing curriculum as a vehicle to assist students understand and comprehend how algorithms can be constructed. Not only that, but both iterative and recursive algorithms for a similar task can be taught parallel, allowing for these algorithms

to be compared for similarities and contrasted for differences. An initial analysis of the data indicates that students perceived the Bubble Sort to be more complex than the Merge Sort and therefore alternative approaches to explain the concept of a Bubble Sort are required, such as a unplugged activity (Feaster et al., 2011) coupled with sound pedagogical approaches such as Peer Instruction (L. Porter et al., 2016a), (B. Simon & Cutts, 2012), (Zingaro & Porter, 2014b). This hypothesis will be evaluated in further studies as well as the analysis of the students' open-ended responses will be analyzed and reported on.

In further studies, the animations with the formative assessment tools will be used to study the effectiveness of a student-centered approach to teaching sorting algorithms by means of scaffolding formative assessment compared to a more traditional style of knowledge transmission within formal lesson settings. These further studies will allow us to validate the protocol and compare different models for teaching and learning sorting algorithms using both recursion and iteration. Further activities involving recursion with engaging games and using real life dataset such as the one proposed in (McQuaigue et al., 2020) will be explored. Additionally, longitudinal studies across a sequence of CS courses could also be undertaken to obtain an insight into the students' cognitive process as they fully understand and master recursion. We anticipate that participants in this longitudinal study would have a starting point of a CS0 course, continue to participate through studying CS1, CS2 and conclude their involvement by completing a first software development course.

4.6 Curriculum on Foundation of programming for talented youth

A curriculum was designed for a three weeks intensive course for talented students.

4.6.1 Foundation of programming: syllabus

4.6.1.1 W1: Design and implement an imperative and procedural program

Day	Time	Competencies	Resources
Mon	9-noon	Honor Code & Computer Use Demographic survey Self-esteem/self-efficacy survey Computing attitude survey Design tool: Natural language, Flowchart, pseudocode, mathematical languages Block based languages	Discuss Honor Code/Expectations Discussion of attitude survey Sum of first/N Numbers with Flowgorithm Conversion of AAA to BBB using a TM Scratch, App Inventor, Snap! & dialect, Python, Java
	1-3 PM	Procedures and Function	Draw figure using Flowgorithm
	7-9 PM	Game time	Lightbot
Tues	9-noon	Loops, nested loops and functions Intro to variables and conditionals Intro to parallel loops	Calculator using increments, decrements and zero assignment: sum, sub, mul, div, pow in N using loops
	1-3 PM	Analysis of algorithms: counting steps and space	Calculator using increments, decrements and zero assignment: sum, sub, mul, div, pow in N using functions. Simple and compound interests
	7-9 PM	Puzzle time: Greedy algorithm	Four people, a lamb and a bridge Huffman code
Wed	9-noon	Variable: characteristics, roles, operators, and expressions	Educational game: learning addition and subtraction
	1-3 PM	Algorithm animations: logo & turtle	Animate the Educational game
	7-9 PM	Puzzle time: Invariants: Parity Coloring Other invariants	Break a chocolate bar, Color of the last marble, A chess knight in a corner to corner journey Counting the number of 3x3 chessboards in an 8x8 one Domino, tromino, and tetromino tiling
Thurs	9-noon	Conditions and logical formulas Propositional calculus Logical deductions	Maximum and minimum of N numbers Logical equivalences of propositional calculus Puzzle time: Logical Deductions and Deduction Trees
	1-3 PM	Well-formed formulas of propositional logic and parsing trees	Puzzle time – Alice in Wonderland Puzzle time: inference
	7-9 PM	Humanitarian Free Open Source Software	Project brainstorming

Friday	9-noon	Working with one dimensional data: list, finite array, sets, maps String manipulation and intro to regular expression	Conversions among number representations Educational app: learn a foreign language Audio manipulation
	1-3 PM	Working with two-dimensional data: list of lists, matrices	Image manipulation
Sund.	7-9 PM	Puzzle time: step counting Game time: Cargo-Boot Puzzle time: backtracking	Tower of Hanoi Cargo-Bot game The n-queens problem

4.6.1.2 W2: Recursion, Data structures and Object Oriented Programming

Day	Time	Competencies	Resources
Mon	9-noon	Pre-test on recursion Recursion Head Recursion Tail Recursion Tree recursion and memoization Recursive equation	Max and Min of N number recursively Matplotlib and graphing functions Fibonacci Number Memoization
	1-3 PM	LIFO data structure	Creating and using a stack Data structure animation Project: a scientific calculator
	7-9 PM	Post-test on recursion Puzzle time: induction Decrease and conquer	Proof by induction Guess the number A fake among eight coins Fake coin detection with a spinning scale Figure tracing
Tues	9-noon	Pre-test on searching algorithms Searching algorithms unordered containers	Find first element Find last element Find all elements Iterative and recursive implementation Searching algorithm animation
	1-3 PM	Searching algorithms with ordered containers	Find first element Find last element Find all elements Iterative and recursive implementation Searching algorithm animation
	7-9 PM	Post-test on searching algorithms Puzzle time: Divide and conquer Intro to parallel search	Parallel minimum with one shared variable Parallel minimum with one shared variable per processor

Day	Time	Competencies	Resources
Wed	9-noon	Pre-test on sorting algorithms Selection sort Bubble sort Merge sort Quick sort	Implementing recursive algorithms both iterative and recursive when appropriate Sorting algorithm animation
	1-3 PM	Transforming a recursive program into an iterative program	Animate the sorting algorithms
	7-9 PM	Post-test on sorting algorithms Intro to parallel sort	Parallel sorting techniques
Thurs	9-noon	Queue Lists	Creating and using a queue and a list Data structure animation Creating complex data structures to solve problems
	1-3 PM	Using a list to implement LIFO and FIFO data structures	Project: list of students with their grades in each course
	7-9 PM	Puzzle time: dynamic programming Exhaustive search	Solving dynamic programming problems with a spreadsheet: the knapsack problem. Three Jugs
Friday	9-noon	Classes, Objects	Educational app: learn a foreign language
	1-3 PM	UML class diagram: classes, attribute, methods	Reading UML diagrams Reverse engineering: from code to UML diagrams
Sund.	7-9 PM	How to present in public	Presentation tool HCI Video

4.6.1.3 W3: Design tools: UML diagrams, E/R, EER, and event Programming

Day	Time	Competencies	Resources
Mon	9-noon	Classes and relations between classes Class diagrams Inheritance Contained in Overloading Polymorphism	Project: a comic shop
	1-3 PM	Sequence diagram Overloading Polymorphism	Project: a comic shop
	7-9 PM	Puzzle time: Transform and conquer	Anagram detection Two Jealous husbands Mental arithmetic A stack of fake coins Missionaries and cannibals
Tues	9-noon	Entity relationship diagram Binary Associations: 1-1; 1-N: NN SQL language Introduction to ternary associations	Project: a Database implementation of a comic shop
	1-3 PM	SQL language Visual Basic for Application	How to design a user interface Event programming
	7-9 PM	Course evaluation survey Self-esteem/self-efficacy survey Computing attitude survey	Implementing set operations in SQL
Wed	9-noon	GUI in textual languages Panels, buttons and events Components and action listeners	Comic shop Graphical Interface Design your own project
	1-3 PM	Inner classes Read the generated source code	Comic shop Graphical Interface Design your own project
	7-9 PM	Project refinement	Comic shop Graphical Interface Design your own project
Day	Time	Competencies	Resources

Thurs	9-noon	Project refinement Prepare your presentation	Project refinement Prepare your presentation
	1-3 PM	Project refinement Prepare your presentation	Project refinement Prepare your presentation
	7-9 PM	Project refinement Prepare your presentation	Project refinement Prepare your presentation
Friday	9-11	Pitch presentations	Pitch presentations
	11-12	Party time	Farewell

Chapter 5: Accessibility in the curriculum

Objective of the chapter: provide technologies and tools for accessible teaching and learning.

Approach: narrative review and a case studies as proof of concept

Result achieved and novelty: Offer a set of guidelines for accessible teaching practice.

Significance for the state of the art and the narrative of the work: Offer an on the filed international comparison of accessible educational practices

According to (Cristaldi, 2020) It is important that quality education for all guarantees that all students have access to resources and instructions that enable all of them to fulfil their potential (Ibe et al., n.d.), (Wobbrock et al., 2011). Failure to reach this goal means preventing access to invaluable resources for members of educational communities. To enable Special Educational Need and offer to students with different abilities the possibility to fully participate in the educational experience, it is necessary to:

- 1 Adhere to both the Universal Design in Learning principles (S. E. Burgstahler & Cory, 2010a). (Rogers-Shaw et al., 2018) and Universal Design Instruction principles (S. Burgstahler, 2009b)
- 2 Participate and put effort into one of the 25 ways that the Disabilities, Opportunities, Internetworking, and Technology (DO-IT) is changing the world (Do-I.T., 2017) through its program¹.
- 3 Develop accessible resources. Putting significant effort into developing accessible resources has a positive effect on the whole educational community at the class, school, local community and nationwide level.

Leveraging on a literature review, framed inside the Technological Pedagogical Content Knowledge model (Maiorana, Richards, Lucarelli, Miles, et al., 2019), (De Rossi & Angeli, 2018b), (Maiorana et al., 2017a), related to special education teacher training in section 2, we will report on experiences across Italy in section 3, Alabama in section 4, and England in section 5, reporting lessons learned and best practices. Section 6 will share some thoughts for discussion and section 7 will draw some conclusions and highlight future work.

5.1 Literature review on technologies, pedagogies and content

Many technologies and resources are available. The Daisy consortium provides a comprehensive list of tools and resources for the creation, conversion and validation of accessible publications² (Park et al., 2019b). In this regard we could list some technologies, pedagogies and innovative content resources:

- a) Poet Image description tool (Diagram center, 2015) and their guidelines for describing images, such as the flow chart guideline⁷².
- b) Captioning software like the Caption and Description Editing Tool (Cadet)⁷³, developed by the National Center for Accessible Media (NCAM) at WGBH educational foundation (Foster & Connolly, 2017), (Linebarger, 2000) helps in completing an automated captioned video.
- c) Interactive video transcripts. Interactivity allows for searching the video using text phrases, running transcripts with synchronously highlighted text, and the possibility to click on the text and jump to the selected part on the video (Wildemuth et al., 2003). An open source fully accessible cross-browser HTML5 media player is available on the GitHub repository⁷⁴.
- d) Tactile reading^{75,76} and digital talking book or spoken book (Argyropoulos et al., 2019) with a crowd-sourced volunteer effort in Europe⁷⁷ and around the world⁷⁸. Crowd-sourcing the effort will allow a rich set of books to be available to a wide community beyond dyslexic people and people facing visual difficulties. They can serve as great teaching and learning tools aligning courses with the multimedia requirements of 21st century learners.
- e) Use 3D Printing⁷⁹ for creating tactile experiences.

Develop accessible hardware and software tools that will assist learners both in daily practice activities and in educational activities. Listing all the initiatives here would not be feasible. In the Computing domain we could cite some remarkable examples such as the Integrated Development Environment (IDE) specifically designed for SEND learners (Milne & Ladner, 2019b).

5.2 The Italian master experience

In Italy, a significant nationwide effort has recently been implemented to train in-service, pre-service and student teachers to teach to special students with special needs (Shinohara et al., 2018b) at all school levels from primary through high school. The effort culminated in multiyear educational initiatives spread nationwide with master courses for teachers. The courses, usually run over one or two years, provide in many cases a nationally recognized certification for teachers of special learners.

⁷² <http://diagramcenter.org/specific-guidelines-d.html#44>

⁷³ <http://ncamftp.wgbh.org/cadet/>

⁷⁴ <https://ableplayer.github.io/ableplayer/>

⁷⁵ <https://learningally.org/>

⁷⁶ <https://www.mtm.se/en/tactilereading2017/>

⁷⁷ <https://www.libroparlato.org/> , <https://adovgenova.com/>

⁷⁸ <https://learningally.org/>

⁷⁹ <http://diagramcenter.org/3d-printing.html>

A laboratory on this master course was designed with these goals:

- 1 Provide the learners with information seeking tools and techniques applied to the domain of interest, eliciting the main reference points and information sources in the domain of interest. Provide an overview on who and what is taught around the world (Shinohara et al., 2018b), (Kawas et al., 2019b).
- 2 Provide the learners with Computational Thinking abilities using tools and techniques applied in an interdisciplinary setting to construct software and hardware tools as well as learning resources useful for people with special abilities. Test and get feedback from the students involved in their daily class activities, asking feedback from others in the same community.
- 3 Promote communication, collaborations and networking among teachers. Reflection and experience reports show that peer-lead meetings had a positive impact on the whole educational process.
- 4 Organize the activities around a curriculum delineated by special abilities focusing on technologies, pedagogies and content.
- 5 Favor project based pedagogies in group settings.
- 6 Leverage on tools such as concept maps (Cañas et al., 2004), (Liu et al., 2011), (Novak & Cañas, 2006) as design tools useful for teachers and students.

The main difficulties faced during the programme were due to a severe lack of time for the participants. An overbooked schedule and a tight and tough master program contributed to heavy cognitive overload in all participants, resulting in lost momentum in the educational activities despite the commitment of teachers and the dedication to their students. Self and group reflection using online communication means should be pursued for after-experience reflections. Blended and on-line instruction initiatives should be sought to avoid depleting time, energy and resources needed by teachers simply to reaching the campus location.

5.3 The Alabama experience

Special education classes will educate P-12 Students with a varying range of special needs, intellectual and developmental disabilities. Assistive technology has always benefited students with disabilities (Carpenter, 2015). However, the line between that which is considered high-tech technology has not only faded, but in some cases it is non-existent. With the increased demand for all students to learn technology to succeed within STEAM careers, it is imperative to provide to learners' digital devices to enhance learning. Simple operating instructions, digital citizenship, and how to manage social media while protecting your identifiable personal information are important skills for gainful employment and independent living. (Yamamoto et al., 2014) found the use of video enabled students with disabilities to retain lessons in STEM related courses. Although the STEM curriculum

emphasizes problem-solving, the “creative inquiry model demonstrates a mutually engaged transdisciplinary approach for STEAM learning that intrinsically values the signature pedagogies in art and design education in synergistic relationship with one or more STEM disciplines” (Costantino, 2018), thereby encouraging inclusion, expanding participation, and cultivating persistence (Payton et al., 2017). Further discussion will include:

- 1 A review of videos used in special education classes to assist students in retaining information and skills.
- 2 The use of simulations.
- 3 Integration of digital and assistive technology.

5.5 Discussion

From the above discussions and from the authors’ own experiences, the authors would raise the following points:

- 1 Computing tools, software, hardware and unplugged are all essential parts of teaching practices.
- 2 CT can and should be used by teachers to co-develop with their colleagues simple and effective Apps and tools for solving real-life problems with their students, thereby improving their educational experiences. The developed products need to be tested with the students.
- 3 Involve all class members in the design and development processes, infusing a culture of inclusiveness.

Chapter 6: Community of Practices

Objective of the chapter: provide process and guidelines for a research-based approach for creating and nurturing a peer-led community of practice

Approach: 1) Call for experiences: topics, and grade band; 2) Asynchronous online negotiations on topics among the authors and decision on topics and index; 3) Discussion on a reporting template according to research guidelines such as (McGill et al., 2018); 4) Online sharing of the experience reports which were peer reviewed; 5) Crafting a list of references for reflections and discussions; 6) Drafting of the discussion and conclusion with peer review

Result achieved and novelty: Investigate the process of designing and applying a research protocol for self-reflection activities supported by the educational practices

Significance for the state of the art and the narrative of the work: Provide guidelines summarising the above mentioned research based reflection approach, supported by a community of practice, with a particular emphasis on the teachers' role in guiding the development of their students in reaching a broad spectrum of competencies

Education is recognized as playing a central role in reaching the Sustainable development goals (United Nations, 2015; Bokova, 2017). The centrality of education has been stated since ancient times by prominent philosophers such as Seneca in many of his works, including his "Epistulae morales ad Lucilium".

Various frameworks of intended leadership education have been proposed, emerging from frameworks (Waters et al., 2004; Waters & Cameron, 2007; Kools & Stoll, 2016; Stoll & Kools, 2017) for teachers (Gouseti et al., 2021), and students (OECD, 2019), and from the scientific literature related to leadership development and competencies in the realm of the formal school system (Leithwood, 2021) and CoP, both non-formal (Prenger et al., 2017) and formal (Dovigo, 2010). This analysis will focus on the Scientix CoP (Nistor et al., 2019), which embraces both STEM (Science, Technology, Engineering, and Mathematics), where improvement is considered a matter of urgency (Niewint-Gori & Gras-Velazquez, 2020), and all other disciplines (STE(A)M) (Jiménez Iglesias et al., 2018). In this context, computing CoP (Morelli et al., 2014; Morrison et al., 2012; Sentance et al., 2014), related to small school settings (Mangione et al., 2021) and rural areas (Wang et al., 2021), offer examples of successful experiences with broad contextual diversity. Theory-based instruments, e.g., the inclusive leadership questionnaire (Li, 2021; Crisol Moya et al., 2020), have been developed and validated, suggesting that inclusive leadership promotes a favorable school climate and culture emphasizing high expectations and quality education.

In this work, we will investigate the role of leadership in education, starting from the CoP to arrive at the students who will become tomorrow's leaders. This will be done through the direct experience of the authors.

The experiences will be described through the lens of leadership education and will focus on the developing process and best practices related to leadership at the class, school, local communities, and at the national level throughout the Scientix CoP.

Reflections and suggestions for policymakers and educational stakeholders will conclude the work.

Among the voices from the field presented in (Maiorana, 2020), (Maiorana, 2022) and in (Maiorana, under review) here we summarize

- a) the use of a class site as a tool for reading, reflecting, writing, coding, community building and leadership development
- b) Inclusive computing for digital humanities
- c) A curriculum for digital citizenship, data, and information literacy

After a discussion related to the key lesson nurtured by participating on a Community of practice we will present a curriculum for perspective teachers emerged by an international collaboration with the work going to be submitted to a peer reviewed journal.

A synthesis of the profession work performed inside researcher and educators communities of practices like Computer Science Teachers Associations, Computing at School, British Educational research and a work within the Scientix community of practice addressing the interlink of research and on the field educational practice will conclude this chapter.

6.1 Scientix Teachers Ambassadors for Critical Digital Literacy and Active Citizenship

This work, leveraging previous experience in research and teaching practice collaborations (Panconesi & Guida, 2017), (Maiorana et al., 2020a), aims to:

- 1) Investigate the process of designing and applying a research protocol for self reflection activities on the educational practices covering all the aspects of the Technological Pedagogical Content Knowledge Framework. The research based approach was carried out by a group of teachers, all volunteers, belonging to the Scientix community of practice.
- 2) After an overview of the state of the art on competencies and skills supporting a quality education, we present, as a proof of concept of the application of above designed protocol, to share content, pedagogies, and technologies guiding the design of educational activities and learning resources aiming at sharpening the above mentioned students' competencies.
- 3) Provide guidelines summarising the above mentioned research based reflection approach, supported by a community of practice, like the Scientix project to improve educational practices and offer an "in the field" view of the Scientix ambassadors

response to some of the questions raised in (Cappello et al., 2022) with a particular emphasis on the teachers' role in guiding the development of their students in reaching a broad spectrum of competencies framed around the Critical Digital Literacy framework proposed in (Gouseti et al., 2021).

The central role of education in reaching all the United Nations sustainable goals (UN, 201520015)⁸⁰) as well as the importance of a research based approach to improve educational practices and support teachers communities of practices, like the Scientix one, is well established and recognized (Buisse & Verbeke, 2003). Quality education is the best leverage to guide active students' learning and make them the chief protagonist of their learning path. Recent recommendations and guidelines have emphasised how Science, Technologies, Engineering, Arts, and Mathematics (STE(A)M), the core of the Scientix CoP, can support a more inclusive education, contributing to reducing the number of young people neither employed nor in education and training (NEETs) (OECD guidelines, 2022) and to improving governance practices and society at large.

By presenting both the process, i.e., how the research protocol was designed, refined and applied, and the products, i.e. the research protocol itself, and the educational activities and learning resources designed, carried out, reported and improved according to the above mentioned protocol, the work aims to:

- 1) advance knowledge on the benefit of the research based approach emerging from the shared research protocol and, as proof of concept, experience reports, their evaluation and their longitudinal improvements and application to the whole Italian Scientix community
- 2) provide guidelines and recommendation on how to replicate the research experience among other communities of practices acting at the national () and international level (CSTA) MCSP), (BERA)

The activities, considered examples of experiential learning (Swennen, 2020) and action research practices (Avison et al., 1999), lay the basis for cooperation between researchers, educators, professionals, and their respective institutions, to evaluate the learning resources and teaching interventions by collecting both quantitative and qualitative data.

As the main implications of this work that could be replicated and improved by other researchers are related to the current and future use of proposed research protocol, reflections and guidelines in other projects supporting the educational and the research practices of the authors and the whole Scientix Italian community (Niewint-Gori et al. 2023), (Newit 2023 a), (Newit 2023 a), (Newit 2023 a), (Newit 2023 a), an international comparison with other international communities of practices (CSTA), (MCSP), is and will be carried out, aiming to facilitate research based collaboration among teachers' CoPs.

⁸⁰ United Nation, 2015, Transforming our world: the 2030 Agenda for Sustainable Development <https://sdgs.un.org/2030agenda>

6.1.1 Process and products

The aim of this section is to describe both the process of creating the research protocol, and its implementation and the products created during the activities carried out. In order to collect the experience reports, one of the authors launched a call to the Scientix ambassador who volunteered to engage in a research-based reporting activity to improve their research and teaching practice while nurturing the seed of a self-directed small community of research engaged teachers. The authors gathered together, using online communication means, at the end of the 2019 with the aim to sharpening their educational practice through research based educational reflections.

Building on the experience reported in (Maiorana et al., 2022; Maiorana et al., 2020b), the process followed the following steps inside the self-selected sample of experienced teachers, all volunteers, in the Scientix community of practice:

- 1) Call for experiences: topics, and grade band
- 2) Asynchronous online negotiations on topics among the authors and decision on topics and index
- 3) Discussion on a reporting template according to research guidelines such as (McGill et al., 2018)
- 4) Online sharing of the experience reports which were peer reviewed
- 5) Crafting a list of references for reflections and discussions
- 6) Drafting of the discussion and conclusion with peer review

Considerable importance has been attributed to point 6, the activity of collaboration, cooperation and revision between teachers. We know from research that reviewing someone else's work can be a powerful learning mechanism. Furthermore, the exchange of good practices and the comparison of pedagogical ideas with peer colleagues is an effective way to evaluate and develop one's own practice and recognize different points of view. This has been demonstrated at all levels, from research to daily teaching practice, especially in high school and tertiary education (Finkenstaedt-Quinn et al., 2021), (Bangert-Drowns et al., 2004). Writing activities are examples of STEAM collaboration and their positive effects go far beyond the mere discipline to include Personal and Social Development (Anderson et al., 2015). Scalability to large classes through the support of Artificial intelligence has been recently addressed (Davies et al., 2021).

As a result of the review process nurtured by the above mentioned activities of collaboration, cooperation and revision, the six operative phases have been enriched by combining and refining the previous sequence of steps with other quantitative and qualitative data collection methodologies. In particular, a semi-structured interview was drafted and shared with the other authors, all volunteers, for feedback and convergence to a consensus. The following semi-structured interview was then shared among all the authors to guide the reshaping of the experience reports according to a given format.

- 1) Why was the experience proposed?

- 2) Which was the context and what were the motivating factors?
- 3) How was the experience designed in terms of content, pedagogies and technologies?
- 4) How did the experience fit the selected reference framework?
- 5) How was the learning experience field tested? How was the assessment conducted, e.g., formative, summative? What were the results?
- 6) How was the experience evaluated? Was the experience longitudinally evaluated? Was the experience proposed in other editions?

The semi-structured interview required a great expenditure of time, to refine the collection first of all of information of a general nature (step 1), and secondly of perceptions on the topic, which led to the start of an effective negotiation of intent allowing for an aggregation of the topics and projects according to common themes and domains (step2).

The themes were grouped according to the following macro areas:

- 1) Information Literacies related to online inquiry process, source selection, validation and verification, in short, learning how to think, not what to think.
- 2) Data Literacy and its importance in achieving the above mentioned first goal and for an active and critical participation in society.
- 3) Digital Citizenship engagement and sustainable use of technologies also related to Computational Thinking.

The semi-structured questionnaire was to be administered asynchronously, as was the discussion on each response. A synthesis of the answers will be presented as a proof of concepts and will be detailed in a successive step as will questions related to “what it is expected from a reporting model”. The whole process related to designing, conducting, analysing and reporting data collected through the semi-structured interview was supported by guidelines obtained from the literature, related to both the scientific (Hove, 2005) and humanities domains (Oplatka, 2018), in order to co-construct and use a 360-degree report template (step 3).

The phase of online sharing of experience reports with related peer review (step 4) was accompanied by a subsequent step, a focus group, thanks to which it was possible to re-elaborate one's own experience, including evaluating data in a different light following peer feedback. Many ideas emerged for the phase of reflection and discussion (step 5). From here the final phase of drafting the discussion and conclusion (step 6) was constituted as a fundamental moment of restitution and sharing among peers. In line with this reflection, it was interesting to have individual occasions in which the authors reflected on their report and developed future steps that they could or would like to implement to improve the quality of the practice itself, after all the ideas obtained from the peer feedback and from discussions during the focus group. These instances were analysed through written reports.

Leveraging on previous experiences, including those nurtured in the organisation of a national conference for the whole community of practice (Niewint et al., 2022) where more than 60 experience reports from teachers spread throughout the whole national territory and covering all the grades in primary and secondary school from K1 to K13 were presented, this work will summarise the teaching experience of a representative sample of the Scientix community of practice. The same above mentioned self-reflection approach was used for the national Scientix conference where extended reports and self reflections of the conference presenters have been collected, and it is planned to share these among the whole Italian teacher community as open resources (Niewint, 2022a), (Niewint, 2022ab (Niewint, 2022c), (Niewint, 2022d), (Niewint, 2022e). The activities presented and the contents selected are fully intellectually justified on condition that they maintain an outlook that recognizes and conceives the existence of the interconnections of knowledge, skills and competencies with solidarity (Morin, 2000).

6.1.2 Scientix ambassadors on the field experiences

The work (Maiorana, n.d.) reports about Scientix ambassadors' in-the-field experiences framed inside the Critical Digital Literacy framework for schools (Gouseti et al., n.d.) which is organized around 8 domains: Technology use; Data Literacy; Information Literacies; Digital content creation; Digital Teaching and learning; Digital citizenship; digital wellbeing & safety; digital communication & collaboration. The experiences and the results describe the efforts of the ambassadors during the pandemic and their daily actions in coping with their research and teaching actions with mutual support of each other in a lifelong experiential learning approach. In particular:

- For the information literacy strand:
 - section 6.1: Computational Thinking (CT) (Lodi & Martini, 2021) , a sub-domain of technology use, will be presented in the context of primary education (stages 1 to 5) with an emphasis on the state of mind sharpened by applying CT in daily life as a way to sharpen young children's ability on how to think
 - In section 6.2: hydroponic greenhouses and integration of coding and CT framed in the context of upper secondary schools (stages 9-13)
- For the data literacy strand:
 - In section 6.3, a training course for high school teachers of Biology, Earth Science and Chemistry and how the activities can be enhanced and deepened through the support of open public data collected with an online inquiring approach
 - In section 6.4, the use of Scientix and the Europeana project in STEAM education framed in the context of lower secondary education (stages 6 to 8)
- For the digital citizenship strand:

- In section 6.5, a climate change module and its relation to digital civic engagement, a sub-domain of digital citizenship, framed in the context of primary education
- In section 6.6, a school project in sustainability and digital civic engagement, a sub-domain of digital citizenship, framed in the context of primary and lower secondary education (stages 1-8)
- In section 6.7, tinkering and coding for renewable energy: investigation into solar hot air collection (stages 6-8)
- In section 6.8, the design and development of learning resources (LR) related to: source validation and verification, a sub-domain of information literacy; open and big data, a sub-domain of data literacy, and their importance for digital civic engagement; sustainable use of digital technologies, a sub-domain of digital citizenship, wise use of time and its relation to digital well being; and kindness, a sub-domain of digital communication & collaboration. The LRs are framed in the context of the last year of low secondary schools and the first two years of compulsory upper secondary schools (stages 8-10).

In the following we will focus on the design and development of learning resources (LR) related to: source validation and verification, a sub-domain of information literacy; open and big data, a sub-domain of data literacy, and their importance for digital civic engagement; sustainable use of digital technologies, a sub-domain of digital citizenship

Teaching resource on information and data for teachers and their students

6.1.3 Designing learning resources for digital citizenship, data, and information literacy

Quality education for all should be supported by learning resources that are able to attract and retain all students offering them a rich set of low floor and high ceiling activities (Maiorana, 2019) spanning different domains, covering different topics and approaching education with a truly interdisciplinary and systemic approach that supports students in honing their talents and transforming their weaknesses into strengths. The experience emerged from the aspiration of the authors to craft and share learning resources with a wider public than the one their students met in class.

6.1.3.1 Context, motivating factors and mapping to the CDL framework

The experience regards a project carried out with the collaboration of a leading international publishing house which involved a group of teachers and researchers in the development of a book related to digital education for teachers and their students in secondary education, with particular emphasis on grades 8 to 13. The publishing house has freely distributed the book to all the teachers of STEM disciplines in the Italian territory⁸¹. Reflection on the

⁸¹ <https://it.pearson.com/pearson-scienze/corsi-secondaria-2-grado/consapevoli-in-rete.html>

experiences can be found in (Cristaldi et al., 2022). A wealth of resources related to critically conscious computing in the realm of secondary education can be found in (KO et al., 2021).

Mapping to the CDL framework

The developed learning resources encompass all the domains of the framework where, according to the overview of the CDL framework, this embrace is understood as aiming to “capture the bigger and more complex picture of critical digital literacy as well as the complexity of the educational practice” in all areas of teaching.

6.8.2 Methods: content, pedagogies and technologies

During the experience, the above mentioned protocol was used in:

- 1) The call for experience which was launched in August 2021 by the chief editor of the publishing house who managed to invite a set of experts from all over Italy, including two authors of this work, The grade band of the audience was indicated.
- 2) The Asynchronous online negotiations on topics among the authors and decision on topics and index which were carried out in collaboration with the authors and managed by the chief editors and their team
- 3) Discussion on a reporting template which was carried out during the negotiation phase in a three month period with biweekly meetings with the authors.
- 4) The learning resources which were crafted by the authors and submitted for reviews by the editorial team and by a blind external review managed by the publishing house according to necessity. The drafting and reviewing process lasted another three months.
- 5) Alongside content, pedagogies and technologies were discussed with the editorial team through written reports during the drafting of the learning resources, the camera ready preparation and during the whole distribution phase, allowing the implementation of the sixth step of the protocol during the following four months
- 6) Finally the sixth step for discussion and refinement was conducted during the next year, exploring improvement and collaboration possibilities.

6.1.3.1.1 Content

The experience regarded the design and development of three learning resources related to:

- 1) Source validation and verification and the online inquiry process
- 2) Open and big data and their importance in developing competencies in searching, selecting, interpreting, analyzing and visualizing data sources for active digital citizenship
- 3) Ecological footprint of Information and Communication Technologies and their ecological use
- 4) Behaviours: how we use our time on the web

5) Why it is important to share kindness

6.1.3.1.2 Pedagogies

The learning resources (LR) were designed according to the following principles:

- Leverage on reviews and state of the art in research. i.e. supporting educational practice with research
- Use-case approach: adapt the same topic and learning path to different audiences: a) From lower to higher secondary school; b) Formal, non-formal and informal education
- Pre-test and post-test
- A flipped content approach (Maiorana, 2020): when possible, the topics are introduced using animations and short tests to guide student reflection. At the end of these preliminary steps, students are able to describe the content of the animation, including referencing technical details, and are ready to engage in meaningful class discussion, reflection and a deepening of their knowledge and understanding.
- Rich set of student-centered assessment activities, both formative and summative
- Solutions with a focus on the process rather than on the solution
- Rich set of references and web links

The LR and the associated activities have been designed with a flipped content approach in mind (Maiorana, 2020), facilitating student-centered approaches such as problem and project based learning. The plan is to involve teachers in the evaluation of the impact of the LR on the student learning path towards digital literacies in order to distill best practices to maximise the impact on students and develop their talents.

Other LRs related to the Digital selfhood and Digital overexposure sub-domains of Digital wellbeing and safety described in (Gouseti et al., n.d.) are under development.

6.1.3.1.3 Technologies

The set of technologies suggested either directly, or implied, is ample, e.g.

- 1) Source validation: from the use of search engines to auto generative networks in artificial intelligence
- 2) Data: data visualization and analysis tools from spreadsheets to coding and databases with textual and visual block languages highly recommended for mastering advanced concepts such as parallel analysis of large volumes of data through maps and reduce primitive (citare Snap!)
- 3) Ecological footprint: exposure to all the technologies mentioned in the text and applications of tools spanning data retrieval, analysis and reporting to chemistry⁸²
- 4) Time on the web: from data analysis to communication, collaboration, presentation and concepts organization⁸³ tools

82 <https://www.chimicaconimattoncini.it/>

83 <https://cmap.ihmc.us/>

5) Kindness: from blog and online communication tools to artificial intelligence software

6.1.3.2 Assessment, impact and conclusions

With a longitudinal collaboration approach, the initiative was renewed by the publishing house with a book designed and developed with a similar approach and framework centered around sustainability issues where the learning resource on the ecological footprint of information and communication technologies was repurposed.

Experience evaluation and impact

According to data kindly shared by the publishing authors, the book was distributed to three thousand teachers. Assuming an average student population of 50 students per teacher, the initiative has the potential to reach fifteen thousand students each year.

Conclusions

The authors advocate for an interdisciplinary and systemic view of teaching that is able to combine humanities and technologies. Only through true collaboration between different areas of expertise, e.g. the teaching and the studying of natural and programming languages, (Fedorenko, 2019), can the cognitive basis of the educational process as a whole be embraced.

6.1.3.3 Discussion

The experience reports are characterized, according to the TPCK framework (Archambault & Barnett, 2010) by innovations in content, pedagogies, and technologies. The content innovation led to exposing students to real life problems in all school grades. According to the OECD legal instrument on Recommendation of the Council on Creating Better Opportunities for Young People⁸⁴, the above-mentioned experiences can be considered examples of youth engagement with global challenges such as climate change and digital technologies.

The innovations in pedagogies are characterized by an Inquiry Based approach typical of the Scientix community, coupled with a problem, project and challenge-based approach. But above all, creating a class climate of mutual care (Maiorana et al., 2022) is considered the best pedagogical approach which, with a consistent investment of time and energy, contributes to the development and promotion of social inclusion and youth well-being, both considered building blocks in the OECD recommendations. This pedagogy of mutual care among the whole learning community is the best way to nurture a kind approach to the communication process. thus improving communications at all levels with all means, as proposed in the commitment to shared responsibility of the Manifesto of Non-Hostile

⁸⁴ <https://legalinstruments.oecd.org/en/instruments/OECD-LEGAL-0474>

Communication⁸⁵. Examples in these directions have been proposed to teachers and their students and can be found in⁸⁶. Developing a kind approach to conflict resolution is considered the best strategy and instrument to support young people's engagement in social action.

The use of cutting-edge technological tools spanning communications, collaboration, learning scenario design, and domain specific needs typical of the discipline, is considered important in facilitating the learning dialogue beyond formal class time and space.

The main lessons learned, and guidelines are:

- 1) a self and peer research based reflection about the educational practice has the potential to greatly enhance the impact on student active learning
- 2) a systemic view of the educational practice embracing all the competencies and a true collaboration among educators from different domains should be sought to achieve efficacy and efficiency. Through interdisciplinarity and multidisciplinary, learning resources and curricula have the potential to address real world problems attracting students' interest. Involving corporations, medium and small enterprises (Maiorana, 2022, special issue introduction) acting at the international, national and local level in designing learning resources combining national guidelines and farsighted job needs has the potential to motivate students' learning ensuring they have the chance to work towards finding a job and sustaining their studies, thus contributing to reducing the number of NEETs.
- 3) The above mentioned collaboration in designing, developing and deploying educational activities should be sought for all the basic literacies, including computing. Designing, developing, deploying, assessing the resources and educational intervention at the department level in collaboration with nearby educational institutions (e.g. middle and high school or secondary and tertiary education) and local enterprises conjugate efficiency and efficacy and facilitate the adoption of different perspectives. Examples in this direction are frequent both in secondary and tertiary institutions. In the computing domain, suggested as facilitating students attraction and retention, this is done at the whole school or university level. Computer science specialists and domain experts from each department design and develop computing courses with practical applications in different domains and deploy the course to all the students and all departments, including through the use of a creative and theatrical approach (Malan, 2023).

⁸⁵ <https://paroleostili.it/en/manifesto/>

⁸⁶ <https://it.pearson.com/content/dam/region-core/italy/pearson-italy/pdf/scienze/CONSAPEVOLI-IN-RETE-PDF-articolo1.pdf>

6.2 The Scientix CoP

The Scientix project⁸⁷, now in its fourth reiteration, is supported by the European Commission H2020 and is coordinated by the European Schoolnet (EUN). Scientix promotes and supports European-wide collaboration between STEM teachers, pedagogical researchers, policymakers and other STEM education professionals.

The activities are coordinated by the European network of national coordinators⁸⁸. In Italy, the coordinator is Indire, the National Institute of Documentation and Educational Research⁸⁹. Scientix Ambassadors⁹⁰, members of the Scientix Teaching Committee, promote and inform their peers from across Europe. They present Scientix in national schools and teachers' associations, at conferences and workshops and can advise teachers on how to collaborate at the European level in the STE(A)M sector. They also help develop and test the various Scientix project tools and services and ensure the pedagogical quality of the Scientix archive.

6.1.1 A class site as a tool for reading, reflecting, writing, coding, community building, and leadership development

The high school experience with a site serving students' needs at the school level has shown a pattern of acceptance similar to the one described in Taylor (2018), with a shift from a one class knowledge and persuasion, to site deployment during the first year. The successive three years were dedicated to the implementation phase. The experience spread to all the classes taught by the author at all grades from K9 to K13. Each class site was used by the class members alone with a steady increase in students' participation, self-determination of the learning path, agency, and awareness of talents. A class of 15 students generated more than 24 thousand page views. The following two years showed a confirmation phase with the students in the previously mentioned class, since they left school, continuing to use the above-mentioned site as a reference for their undergraduate studies and first professional experiences, generating more than 6 thousand page views. They, acting as leaders, accepted the teacher's invitation to act as role models and near mentors for enrolled students.

A site-based approach was used in outreach activities at the local level involving teachers, educators, and informatics professionals.

At the international level, the approach allowed for teacher-led crowdsourcing of educational resources and assessment activities (Giordano, 2015). The last three years, once the pedagogical and technological practices were accepted by the school and the local community, were characterized by an increase of student agency and self-determination. This allowed for a steady decline of the involvement of the author, with a better balance between care for the students and self-care (Rose & Adams, 2014).

⁸⁷ <http://www.scientix.eu/home>.

⁸⁸ <http://www.scientix.eu/national-contact-points>.

⁸⁹ <https://www.indire.it/en/>.

⁹⁰ <http://www.scientix.eu/in-your-country>.

Student-centered, constructivist pedagogies rooted in pedagogy of care were actioned through a great amount of quality time devoted to the effort. The site allowed for collective leadership (Sergi et al., 2012; McCauley & Palus, 2021) where all the individuals were considered active participants *in* leadership, but not containers *of* leadership. The class site allowed to enact actions in many domains of leadership practices in education up to senior public servant (Leithwood, 2012; Leithwood, 2021; Gerson, 2020):

- concretizing change actions which characterize leadership as opposed to management which focuses on the *status quo*;
- sharing short, medium, and long-term goals. The shared vision can be used in a longitudinal way outside class time and space;
- nurturing a high-performance expectation from the self, the members of the group and the members of the whole community. The high-performance aspirations are simply: develop your talents to your best;
- building relationships, developing people, creating “networked collaborations” beyond the class and outside the own organization, building a collaborative culture. Each member of the learning community contributes with questions, answers, activities, resources, technologies reflections and best practices;
- building a sense of internal accountability through setting reachable demands, i.e., in the Vygotsky’s zone of proximal development (Borthick, 2003), and meeting this expectation for the self, the group and the whole community;
- openness to inclusion by “challenging their own perceptions”, looking for and listening to the perspectives of others;
- nurturing a sense of organizational stewardship in all participants, by reinforcing a trust- and value-based culture.

6.1.2 Inclusive computing for digital humanities

A three-year experience was conducted at the University of Catania Department of Educational Science during the academic years 2016/17, 2017/18 and 2018/19 in a Computing and Information Technology courses for third year undergraduate students in a Tourism Management Studies program. The undergraduate course was mainly focused on humanities disciplines with the computing course the first and only such course in the three-year long program of study. For most of the students, with a background in classical studies at high school, the university computing course was the first one in their life. The average number of enrolled students in each course was 70, 85% of which were women and 25% of the overall student population was late in their program of study, giving a starting number of students attending the first lesson of the course 50. The number of attending students remained almost constant during the courses when care was taken to ensure an equal spread of lessons over the semester and a careful choice of the date of the midterm and final

exam. The midterm and final were voluntary exams which substituted the written part of the exam and most of the oral part. Besides these midterm and final exams, there were nine regular exam sessions available to the students during the academic year. Students greatly appreciated the midterm and final exams. Nine sessions of exams represent a great difference between the Italian educational system and other educational systems like the one in the USA where the final exams are one-shot on a pass/fail basis. Student assessment was mostly based on projects demonstration in front of peers with an oral presentation and question/answering session. The courses were held during the first half of the third year, between October and January, with 24 two-hour meetings each for a total of forty-eight hours in two weekly meetings. The objectives of the courses were: learning to learn (European Commission), learning how to do (Unesco), being able to use digital tools interactively (OCED), acquiring literacy in information, media and ICT (Partnership for 21st century skills), how to communicate and collaborate using new technologies (Griffin & Care, 2014), develop an aptitude for Problem Solving and computational thinking (Wing, 2006). To hone problem solving and Computational Thinking skills, App Inventor (E. Patton et al., 2019) was used as the main tool to present basic programming constructs (Nardelli, Forlizzi, Lodi, Lonati, Mirolo, Monga, Montresor, et al., 2017a): variables operators and expressions, conditional statements, loops, procedure and functions, and structured data like vectors and matrices presented in different learning trajectories (Izu et al., 2019) according to the main project developed during the course. Course projects size and complexity increased during the various course editions thanks to the use of cloud shared project repositories managed by the instructor and shared with all the students. In particular, at the beginning of the second and third year the best project of the previous course edition was presented to the class, but the best fly-wheel for this process was peer-student-led collaboration improved by group project development suggested during the course. To the rich set of components used by the students, and in addition to the type of service offered by the apps that had been developed during the first two course editions, during the last year of the course it was decided to develop and add a Yandex translator to the project, in order to provide a translation and language learning app for tourists. Some students got the idea from this to develop a prototype app to translate from the local dialect to Italian and then use the translator with a foreign language. Another addition to the course was the presentation of a sorting algorithm, both iterative (bubble sort) and recursive (merge sort), through animations and three formative tests guiding and scaffolding students through algorithm comprehension demonstrating the feasibility of introducing computing concepts. Accessibility issues with similar pedagogical approaches and technological resources were used in a master course for teachers' certification for special education run at the same Department.

6.1.3 A curriculum for digital citizenship, data, and information literacy

The experience regards a project done with the collaboration of a leading international publishing house which involved a group of teachers and researchers to develop a book related to digital education for teachers and their students in secondary education with particular emphasis to grades from 8 to 10. The publishing house has freely distributed the book to all the teachers of STEM disciplines in the Italian territory⁹¹. Reflection on the experiences can be found in (Cristaldi et al., 2022). A wealth of resources related to critically conscious computing in the realm of secondary education can be found in (KO et al., 2021). The experience relates to the design and development of three learning resources related to:

- 1) Source validation and verification and the online inquiry process
- 2) Open and big data and their importance in developing competencies in searching, selecting, interpreting, analyzing and visualizing data sources for active digital citizenship
- 3) Ecological footprint of Information and Communication Technologies and their ecological use
- 4) Behaviours: how we use our time on the web
- 5) Why it is important to share kindness

The learning resources (LR) were designed according to the following principles:

- Leverage on reviews and state of the art in research
- Use case approach: adapt the same topic and learning path to different audience: a) From lower to higher secondary school; b) Formal, non-formal and informal education
- Pretest and post-test
- A flipped content approach (Maiorana, 2020): when possible the topics are introduced by animations and short test to guide student reflections. At the end of this preliminary steps students are able to describe the content of the animation, also in respect to technical details and are ready to engage in meaningful class discussion, reflections and deepening process.
- Rich set of student-centered assessment activities, both formative and summative
- Solutions with a focus on the process rather than on the solution
- Rich set of references and web links

The LR and the associated activities have been designed with a flipped content approach in mind (Maiorana, 2020) facilitating student centered approaches such as problem and project based learning. The plan is to involve teachers in the evaluation of the impact of the LR in the student learning path towards digital literacies in order to distill best practices for maximizing the impact on students and developing their talents.

Other LRs related to Digital selfhood and Digital overexposure subdomains of Digital wellbeing and safety described in (Gouseti et al., n.d.) are under development.

⁹¹ <https://it.pearson.com/pearson-scienze/corsi-secondaria-2-grado/consapevoli-in-rete.html>

6.1.4 Discussion

From the above experiences it is evident that, as Freire put it, «education is an act of love». Building an inclusive learning community is the best way, both inside and outside the classroom, guiding students to express themselves. The community building process is intended among peers with a distributed leadership model. Guidelines and best practices, summarizing two years of community reflections (Maiorana, 2020), for developing leadership are:

- encourage a climate of trust and self-awareness of the unique talent each one brings to the community. With an inquiry-based approach, a good icebreaker activity could be a reflection on “Who fails the most: the ones who do nothing, or the ones who act striving for excellence?”, in other words, please don't be afraid of mistakes! Encouraging self-reflection on accomplishments during the learning path should be suggested as frequently as possible;
- prefer a project-based approach. Letting learners choose their projects fosters intrinsic motivation (OECD, 2021);
- let the project be as inclusive as possible, i.e., propose activities involving different talents and let the students choose. Encourage the effectiveness of this practice applied to humanities, scientific and technical projects. Combining humanitarian efforts helps to increase motivation (Hislop & Ellis, 2017);
- favor a participatory culture where the learners produce their artifact. Theater, usually considered a high culture domain, has been successfully leveraged in science, too;
- support the activities of the community with guidelines and research-based methods by reflecting on the process to arrive at the product. Examples in this direction can be found from transmedial, multimedia and instructional principles (Mayer, 2020) to bibliographic, information retrieval and fact checking (McGrew, 2020); reflecting on the importance of proper citations to support and highlight personal work naturally leads to self-triggered plagiarism avoidance by the students themselves. Sharing age-appropriate reading on social science theories like the pedagogical approach proposed by Freire (Freire, 2013), nowadays revitalized in Ko (Ko et al., 2021). Social theories in computing have been reported in Cristaldi (2022), too.

The more everyone in the learning community is emotionally and operationally involved in the realization of the projects, the better the class group is consolidated, and all together participate in the success of the educational experience.

6.3 Pre-service Teachers Formation via Systems Thinking and TPCK from Italy, England, and USA

6.3.1 Introduction

For nearly a decade, there has been a worldwide resurgence to adopt compulsory P12 computing education. This revival shifted focus to computer science, coding and computational thinking to transition learners from consumers to creators of digital artefacts. Different countries are at different stages of adopting computing education with England regarded as an innovator, Italy within the early majority group, and USA as a late adopter. In the USA, states are independently reliant on the federal government's framework of policies and laws, but required to meet the diverse cultural dependencies within each state. By 2018, forty-four states adopted or are in the process of adopting standards in teaching computing education (CSTA K-12, 2017; National Standards, 2019; Richards & Turner, 2019; Code.org, 2018). Most states blended two or more organizations' recommendations (CSTA K-12, 2017; National Standards, 2019; Richards & Turner, 2019; Code.org, 2018; Maiorana, Csizmadia, & Richards, 2020).

Recently researchers around the world have proposed a systemic approach (Fuller & Kim, 2022; Sengeh, 2022) as a way to achieve a holistic development of students (Datnow, 2022). Research (Maiorana & Cristaldi, 2023) has highlighted the importance of a system thinking approach to transform schools (Fuller & Kim, 2022), supporting teachers, the mind and heart of the educational system around the world, through adequate policies supporting their continuing professional learning (Boeskens et al., 2020a; OECD, 2019b; OECD, 2021b), advocating for upskilling and investing in people through bottom-up solutions and insights (OECD/OPSI, 2020).

It is clear that a global perspective on teaching is needed for an ample and systemic reform of the educational system involving all the actors of the educational community and in this respect a wise use of digital technologies can contribute towards a positive digital and green transition of society. According to the European Parliament's decision to establish the Digital Decade Policy Programme 2030⁹², "Digital technologies should contribute to achieving broader societal outcomes that are not limited to the digital sphere, but have positive effects on the everyday lives and well-being of citizens. If it is to be successful, the digital transformation should go hand-in-hand with improvements as regards democracy, good governance, social inclusion and more efficient public services".

In this regard, this work aims to make a small contribution in this direction by providing, as a proof of concept, a set of strategies to overcome identified problematic areas, along with a set of best practices and resources in suitable for teaching computing to prospective teachers, through the lens of system thinking.

This work examines concerted efforts in creating computing education uniformity through various initiatives (Maiorana, Csizmadia, & Richards, 2020; Royal Society, 2017; Alabama Code Title 16 Education, 2019; Forlizzi, 2018; Klopfenstien, Delpriori, Maldini & Bogliolo,

⁹² <https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32022D2481&qid=1687470375262>

2019; Redecker, 2017). For compulsory computing education renaissance to flourish, sufficient competent, capable, and confident professional computing teachers need to be recruited and trained. In this work the authors, drawing upon their experience and expertise as initial teacher educators in different countries, analyze the complex process of pre-service computing teacher transformation through the lens of system thinking.

Similarities are acknowledged, while differences in approaches are highlighted. Identifiable stages for computing educators include recruitment of candidates who are trainable to teach computing, supporting pre-service computing teachers from the classroom to becoming in-service computing teachers. Using the lens of systems thinking, a review of research and theory to examine pre-service teacher (PST) formation will include Technological Pedagogical Content Knowledge (TPCK) in the realm of science, technology, engineering, arts, and mathematics (STEAM). Celia (Computer Educators Learning Inclusive Actor) will represent the teaching candidates' professional pathway in each country. Figure 1 illustrates the key stakeholders at each stage, thus contextualizing the system where the problem may reside.

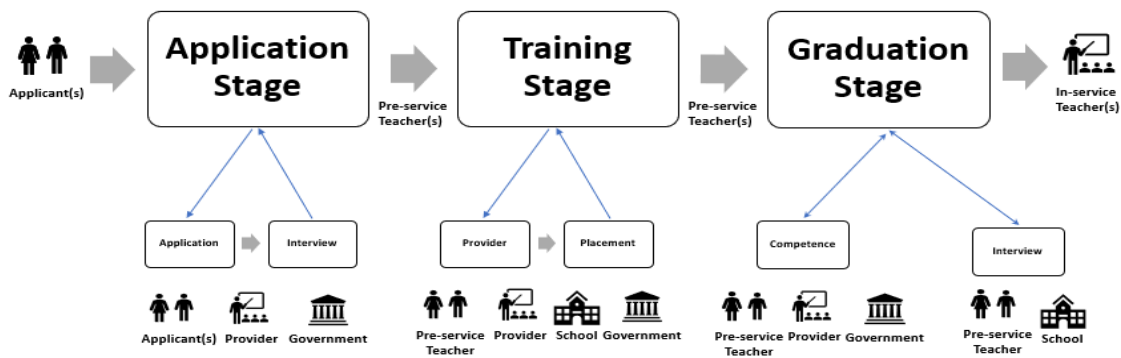


Figure 1: System Workflow of Computing Teacher Formation

Celia (Computer Educators Learning Inclusive Actor) represents the persona of a pre-service teacher as they transit the different phases during the teacher metamorphosis. When Celia applies for a computing educator program, she may be required to take a national or proprietary examination before the faculty assesses her ability to learn methodologies in teaching and computing education. If Celia is conditionally accepted, she must achieve benchmarks based on Specific Measurable, Attainable, Realistic Timeframe (SMART) targets. Table 1 outlines the number of required hours of curriculum in the respective countries or universities and Table 2 reflects the teaching requirements by grade bands.

Table 1 Curriculum in Authors' Respective Countries or Institutions

Country	Year 1	Year 2	Year 3	Year 4	Year 5	Certification Exams
England Secondary: Postgraduate route	Celia must graduate from an undergraduate computer related program where computing forms 50% of the program's content. This is a compulsory requirement to be considered as an applicant for a postgraduate secondary computing trainee teacher program.		Celia studies a one-year Postgraduate Certificate in Secondary Computing Teaching with Qualified Teacher Status (QTS). This course consists of: School placement 120 days Professional Enquiry and Subject Leadership in Computer Science 200 hours Evidence Informed Learning, Teaching and Assessment in Computing 200 hours Transition and Enhancement Placement 75 hours Subject Knowledge and Professional Practice for Teachers 240 hours Once graduated Celia will complete a further two years in a secondary school being assessed against the Early Career Framework.			Celia presents a portfolio of evidence against the Teachers' Standards and passes the academic postgraduate assignments.
England Primary: Postgraduate route	Celia must graduate from an undergraduate program where 50% of the program's content is that of a national curriculum primary subject. This is a requirement to be considered as an applicant for a postgraduate secondary computing trainee teacher program.		Celia studies a one-year Postgraduate Certificate in Secondary Computing Teaching with Qualified Teacher Status (QTS). This course consists of: School placement 120 days Professional Enquiry and Contemporary, Creative and Innovative Practice in Core Curriculum 200 hours Processes, Application, and Influence of Assessment Practices on Teaching and Learning 200 hours The Core Curriculum 100 hours The Broad Curriculum 100 hours Professional Studies in Education 100 hours			Celia presents a portfolio of evidence against the Teachers' Standards and passes the academic postgraduate assignments.

Country	Year 1	Year 2	Year 3	Year 4	Year 5	Certification Exams
			Once graduated Celia will complete a further two years in a secondary school being assessed against the Early Career Framework.			
Italy Primary with CFU	1 CFU = 6 hour for lessons, 19 hours for independent study. In round brackets the CFU for Laboratory activities.					Teaching certificate for primary teachers
	Pedagogy 8 Art 8 Didactic 6 Didactic technologies Physiology for special education 8 Ecology Physics Sport Science Geography History of Education Hygiene	Italian literature Linguistic History and didactic of history 16 Didactics 8 (2) School and clinical Psychology 16 Educational sociology 8 Mathematical logic 8 (1) Mathematics 12 (1) Music 8 (1) Italian Literature and Linguistics 12 (1) English certification 2 (4) Internship and Practicum 11 Thesis and final exam 9				
Italy Secondary	Celia must complete 13 years of school, 3 years of undergraduate courses and two years of master courses in technical domains such as Computing, Engineering, Physics and mathematics according to national laws ⁹³ . With the Master Celia can start seeking non-tenured teaching contracts. After 3 years of teaching services in an 8-year time span she acquires the teaching certificates. Celia can also obtain her teaching certificate after completing an annual university course requiring a 1,500 workload and 60 CFU. With her teaching certificate and 24 CFY on Anthropology, Psychology or Pedagogy she can participate in a national competitive exam with available positions identified according to regional needs. If Celia is ranked among the first positions available in the region where she participated, she becomes a tenured teacher.					

⁹³ DPR 19/2026 and DM 259/2017

Country	Year 1	Year 2	Year 3	Year 4	Year 5	Certification Exams
USA- Alabama Computer Science Educator (6-12)	In the first 2-years Celia will need to successfully complete: Humanities 9 hours Literature 6 hours Fine Arts 6 hours History 6 hours Social/Behavior Sciences 6 hours Sciences 12 hours Mathematics 6 hours <u>Computer Science</u> 9 hours Total Hours Years 1&2: 60 hours		In the last 2-years Celia will need to successfully complete: Professional Studies 9 hours Teacher Ed. Coursework 24 hours Internship 6 hours <u>CS Courses</u> <u>21 hours</u> Total Hours Years 3&4: 60 hours Alabama Code § 290-2- 2-101(2) outlines the required hours for specified teaching fields.		The initial CSE certification can occur in a bachelors (Class B) or masters (Alt-A) program. After the teaching certification is received additional degrees could result in promotion or salary increase.	Praxis 5652 Computer Science Exam and a report in a Case Study format containing Celia's lesson plan(s), activities, a self- reflection, and samples of student assignmen ts to measure her impact and effectivene ss on her student's learning.

Table 2 *Certifications and Responsibilities for Teaching Computing Education*

Country	Grades Pre-K-3	Grades 4-5	Grades 6-8	Grades 9-12
England	Taught by certified primary teachers following the Computing Programme of Study. They may have completed either an undergraduate primary teacher program or a postgraduate teacher training program and have gained a Teach Primary Computing Certificate awarded by the British Computer Society.		Taught by certified secondary teachers following the Computing Programme of Study. These may be certified as PGCE Secondary Computing teachers and have gained a Teach Secondary Computing Certificate awarded by the British Computer Society.	
Italy	Taught by primary teachers with a teaching certificate obtained after 5 years of higher education and after winning a national competitive exam.		When computing is in the curricula it is taught by non-tenured teachers with proper certification or by tenured teachers who won a national competition. To obtain certification teachers must have a master's degree in STEM domain.	
USA- Alabama Computer Science Educator (6- 12)	Taught by educators certified in early childhood or elementary education. Digital literacy and fundamental skills are predominantly taught in these grades.	Taught by educators certified in early childhood or elementary education. Digital literacy becomes a lesser topic and building knowledge and skills in computer science begin.	Currently taught by educators with a secondary education field certification. Currently CS is integrated into all curriculums.	When the Computer Science Educator (6-12) teaching certification is standard practice, Celia will be responsible for teaching programming, algorithms, databases, operating systems, AI, HCI, and much more.

Additionally, the institution will provide Celia an academic environment to explore theoretical and pedagogical approaches in teaching computing under a recognized subject matter expert's guidance and the institution's technological infrastructure. For Celia to become successful, she will need access to a learning, teaching, and assessment environment designed and developed by the experience of a computing educator mentor. The governmental agency is not responsible to provide Celia the competence framework that she will be assessed against but specify the compliance framework that educational institutions need to adhere to in order to be licensed to train teachers (Maiorana, et al., 2020; Royal

Society, 2017; Alabama Code Title 16 Education, 2019; Forlizzi, et al., 2018; Klopfenstien, et al., 2019; Redecker, 2017).

6.2.1 Relationship between Systems Thinking and TPCK

The process of envisioning and thinking of problems and solutions using various systematic thinking styles defines systems thinking (Goodman, 2018). This develops an understanding about processes that are not optimized to permit expansion to create satisfying, long-term solutions to chronic problems. Consequently, systems thinking uses the character traits of curiosity, clarity, compassion, choice, and courage. Because computing relies on systems thinking, Celia must demonstrate the ability and willingness to conduct in depth situational exploration, recognize interrelated characteristics, identify, and test multiple interventions for resolutions including those that are not popular (DfE, 2020; Alabama Code, 2019). Figure 2, and Table 3 exemplifies the relationship between system thinking and the TPCK components. According to (Koehler, Mishra & Yahya, 2007), “At the heart of TPCK is the dynamic, transactional relationship between content, pedagogy and technology. Good teaching with technology requires understanding the mutually reinforcing relationships between all three elements taken together to develop appropriate, context specific strategies and representations.”. This vision has been embraced, according with (Tokuhama-Espinosa, 2023) by learning science where psychology (mind), neuroscience (brain) and pedagogy (education), but also philosophy, cultural anthropology, linguistics, and artificial intelligence provide support for a better understanding of human learning. From the Learning Sciences emerge transdisciplinary insights into the Science of Learning whose end goal is improved teaching.

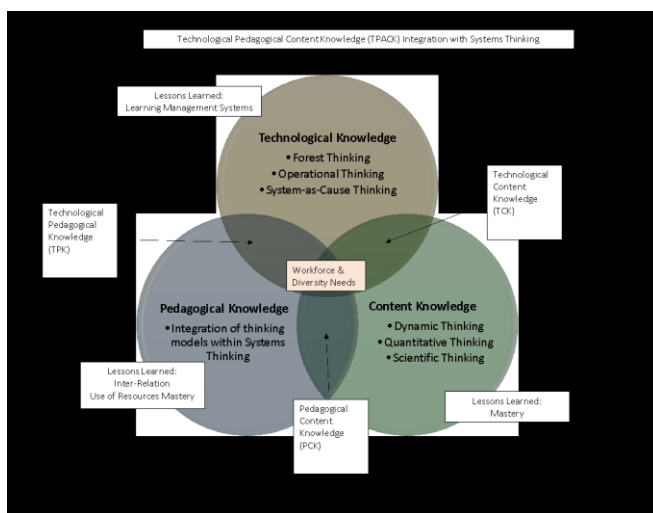


Figure 2 TPCK Integration of Systems Thinking Styles

Table 3 System Thinking Styles Relationship to TPCK Categories with Definitions

TK Thinking Style	Definition
Forest Thinking	Sees beyond the details to the context of relationships which are embedded.

Operational Thinking	Thinking in this stage allows logical reasoning
System-as-Cause Thinking	Known as Endogenous Thinking sees internal actors that manage policies of the system is responsible for the behavior.
CK Thinking Style	Definition
Dynamic Thinking	Ability to make optimal decisions in a dynamic or ever-changing environment
Quantitative Thinking	Analysis and interpretation of real-world quantitative information to draw conclusions and resolve issues.
Scientific Thinking	Involves the content of science and the reasoning process such as induction and deduction, or causal reasoning, concept formation or hypothesis testing.

6.2.2 Candidate Selection.

Countries differ in theory and practice of the selection process. The universities are responsible in the assessment of Celia's aptitude and attitude for TPCK, systems thinking, and STEAM based learning (Maiorana, et al., 2020; Royal Society, 2017; Alabama Code Title 16 Education, 2019; Forlizzi, et al., 2018; Klopfenstien, et al., 2019; Redecker, 2017).

Italy. Table 1 outlines bachelor and Master requirements. For being a primary teacher Celia must undertake an admission test administered by each university following national guidelines (Forlizzie, et al., 2018; Ministerial Decree 214, 2020) enacted in computing curricula (Maiorana, 2019) After completing her master, Celia must secure subject specific teaching concur in order to teach either at primary or secondary level.

England. Celia will apply a maximum of 3 training providers (university-based or school-lead) via an online portal. They are interviewed to determine their potential to train to teach, prior to an offer formally being made. Additionally, the training provider ensures that individual candidates fulfil national compliance requirements prior to commencing their training as part of a rigorous selection process (Royal Society, 2017).

USA. Institutions will vet Celia using multiple measures of scholarly and dispositional suitability (Alabama Code, 2019). Each university bears responsibility candidates have a minimum GPA on earned college credit, pass a state-endorsed background check, and demonstrate appropriate ethical dispositions. Candidates are interviewed specific to each university campus, therefore the standards for selection are uniform while the processes differ.

6.2.3 Technological Pedagogical Content Knowledge (TPCK) Framework

Acquisition of TPCK principles align with STEAM instruction using systems thinking to better prepare global citizens for the many vocational STEAM opportunities. Emphasis on TPCK and STEAM transitions the traditional rote lesson into experiential learning (ALSDE, 2018;

Artworks, nd; Culture Learning Alliance, 2017; Di Blas, Fabbri, & Ferrari, 2018; Maiorana, 2019; Nesta, 2014; STEM Learning, 2020; The Big Draw, 2020; Royal Society, 2020). This section explores the introduction, impact, and challenges of TPCK-enhanced STEAM instruction.

Italy. Celia attends training courses for primary education related to implementation of the TPCK framework (DeRossi & Trevisan, 2018, Maiorana, Richards, Lucarelli, Berry, & Ericson, 2019) and STEAM. European projects offer a wealth of resources related to TPCK (ITELab Project, nd; Blamire, Cassells, & Walsh, 2017). A national master and summer schools⁹⁴ offers Celia updates with technologies (Mandrioli, Torrenbruno, & Marini, 2010). According to Reddecker (2017), in STEAM computing is introduced either as a standalone discipline or embedded in the context of other domains (Caspersen, Gal-Ezer, McGettrick & Nardelli, 2018). Proposed Italian national computing guidelines (Forlizzi, et al., 2019) can be used as guidelines for Celia's formation. Recent surveys conclude that pre-and in-service teachers often use informal training (Klopfenstien, et al., 2019).

England. Celia is introduced to the rationale (Culture Learning Alliance, 2017) and TPCK (Nesta, 2014) for developing, supporting, and sustaining a STEAM classroom. National initiatives, such as STEAM Toolkit (Artworks, no date). The Big Draw Festival (2020).

Royal Society's Partnership Grants (2020), support the teaching of STEAM in schools by developing creative and collaborative teaching and learning communities. Impact of STEAM initiatives are articulated by both STEM Learning (2020) and the Royal Society (2020).

USA. Teacher programs adjusted the focus to TPCK and systems thinking content (CSTA K12, 2017; ALSDE, 2018) to address preparation of college and career ready students. The adoption of Digital Literacy and Computer Science Standards (2018) meant integration into the different content areas to push TPCK across the curriculum. Now the platform allow faculty to broaden the boundaries of TPCK as Celia negotiates the content knowledge and pedagogy of her discipline (ALSDE, 2018; Richards & Turner, 2019). She is expected to blend STEAM and TPACK initiatives with systems thinking to provide a robust and real-world skillset to her students.

6.2.4 Internship and Practicum.

Celia's formation and transformation will be explored through the lens of Kolb's Experiential Learning Cycle (1984) as it relates to systems thinking. Although this is the primary lens, the ability to interweave TPCK and STEAM are relevant in the scoring of Celia's capabilities as a co-instructor.

Italy. During Celia's placement, she will complete an average of 500 hours in different schools. This allows her to be engaged with Kolb's (1984) multiple low-stakes practicum experiences related to TPCK and course goals (Lotter, Singer, & Godley, 2009).

⁹⁴ Coding summer school

England. Celia is required to complete 120 days of placement in two contrasting school settings (Royal Society, 2017), teaching, generating, and curating competence-based evidence for her Qualified Teacher Status (QTS) certification (DfE (b, c), 2020). Through this process, Celia develops as a critical reflective TPCK practitioner (DfE (c), 2020).

USA. Alabama Code (2019) states clinical experiences occur at least two times during Celia’s matriculation using various diversity criteria. Field experiences integrate TPCK with 6-12 standards (Alabama Code, 2019; ALSDE, 2018). Immersing Celia in learning communities to demonstrate competence in STEAM content, she will begin to integrate systems thinking, TPCK, and computing into a comprehensive approach, e.g. the iceberg model, to build competencies as depicted in figure 3. Successful internship completion requires Celia to integrate TPCK into all instruction including the final computing education instructional unit with interactive activities. She will present her case study with lesson plans, cooperating teacher and university supervisor observations, self-reflection, and student assignments to measure outcomes to ensure all criteria in planning, instruction, and professionalism set forth by the institution and the state are met.

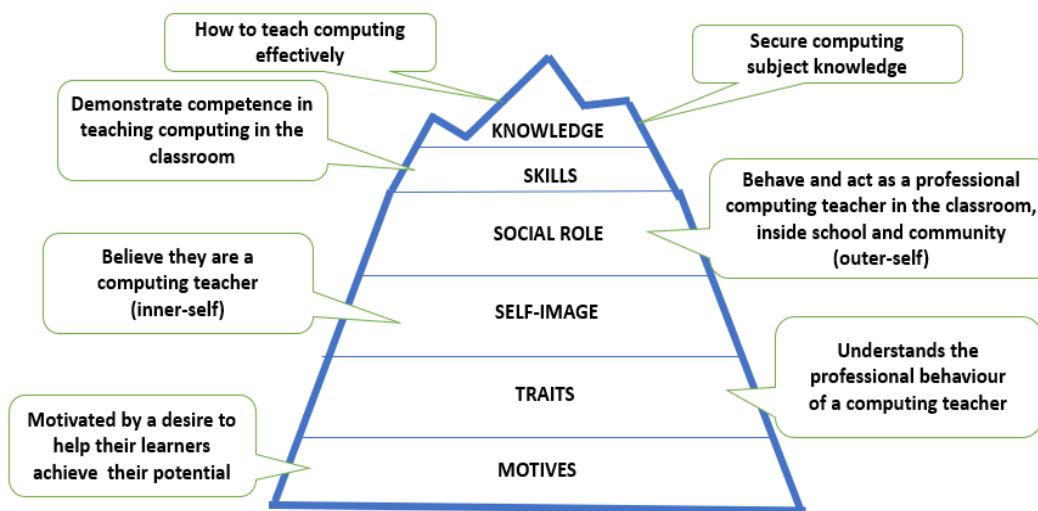


Figure 3: Iceberg Model of Competencies for Teaching Computing

Table 4 Role of TPCK, Systems Thinking and Styles in Computing Education Formation

	Motive <i>A desire to help learners achieve their potential.</i>	Traits <i>Understands the professional behavior of a computing teacher.</i>	Self-Image <i>Believes they are a computing teacher (inner self)</i>	Social Role <i>Behaves and acts as a professional computing teacher in the classroom, school, and community (outer self).</i>	Skills <i>Demonstrate competence in teaching computing in the classroom.</i>	Knowledge <i>How to teach computing effectively.</i>
Thinking Style	Operational	Systems-as-Cause	Forest	Quantitative	Scientific	Dynamic
Technological Knowledge (TK), Pedagogical Knowledge (PK), and Content Knowledge (CK)						
TK	Does Celia address ethical issues and influence in coding HCI, AI and other current topics in computing education?	Does Celia respond to uncomfortable questions effectively on well versed and topics with less knowledge?	Does Celia project her knowledge and self-confidence to her students during instruction?	Does Celia project a respectful teamwork, collaborative environment?	Does Celia demonstrate competencies with age and grade appropriate exercises and content?	Does Celia effectively promote the integration of systems thinking and computing competencies in students?
CK	Does Celia explain	Does Celia address the	Does Celia constructi	Does Celia promote a	Does Celia assist students in	Was Celia well prepared

	digital literacy and other computing concepts as it relates to the content subject area?	negative impact of content knowledge when applied inappropriately among her students?	vely guide and praise students as they work throughout their learning path?	respectful collaborative teamwork, and acceptance of diverse thinking?	learning content, skill, and competencies effectively through positive reinforcement and empowerment?	in assessing students' overall computing competencies?
PK	Does Celia employ systems thinking and other pedagogical, social, and emotional approaches effectively?	How flexible is Celia in managing the computing classroom of varying degrees of student learning in technology?	Does Celia promote a positive sense of identity & self-worth in her students when they struggle in a content or skill area?	Does Celia empower her students to be a digital leader in a respectful teamwork, collaborative environment?	Does Celia effectively use systems thinking in her pedagogical knowledge to effectively develop students' content, skills, and competencies in computing?	Has Celia acquired the instructional strategies to effectively teach computing during periods of outages?

6.2.5 Transiting to In-Service.

Ultimately, Celia's student outcomes will determine her effectiveness as a computing educator. Therefore, the role systems thinking and TPCK in the realm of STEAM education during Celia's transition to an effective in-service computing educator is discussed (Alabama Code, 2019; Dfe 2020a; DfE 2020b; DfE 2020c; Kolb, 1984; STEM Learning, 2020). If Celia is unable to acquire the specified level of competencies in the processes and styles of systems thinking with TPCK, then she may not be able to effectively transfer the knowledge, skills, and abilities to her P12 students.



Figure 4 Scaffold of Systems Thinking and TPCK in Computing Education and STEAM

Italy. During the first year of tenured status, Celia is required to attend an induction program (Mangione, Pettanti, & Rosa. 2016). The program consists of preparatory meetings, training workshops in relation to TPCK, group project, professional mentor supervision, and online training aimed at building a digital professional portfolio.

England. Celia will present her competence-based evidence for achieving each of the 8 standards of the Qualified Teachers' Standards (QTS) professional certification (DfE (a), 2020). She is recommended to the Department of Education (DfE) by her training provider. Upon employment, she will undertake the Newly Qualified Teacher (NQT) induction year. This development is underpinned by the early career framework including TPCK. NQTs are supported by her subject based mentor.

USA. Certification transition includes successful completion of the program, pass required proprietary exam, Praxis 5652, to demonstrate content knowledge and competence in effective teaching, and two field experiences prior to internship (Alabama Code, 2019). Celia can request additional instruction to be successful during her first two years in the classroom. The university is required to provide assistance including technology integration and TPCK skillsets appropriate to the teaching field.

6.2.6 Lessons Learned

Figure 1 Systems Diagram, discussed in the introduction, outlined the key stakeholders and the four key stages in teacher transformation. This diagram illustrates the interaction and interdependence between different stakeholders at each stage of teacher transformation. After Celia becomes an in-service teacher, she may voluntarily decide or have the decision made for her to leave the profession, which is a critical failure in the teacher preparation process. Possible reasons for leaving could include academic progress, attitude, aptitude, achievement against teachers' competence standards or a combination of the aforementioned. The potential failures and strategies for resolutions Celia could experience are summarized in Table 5.

Table 5

Dispositional and TPCK Challenges During Computing Educator Formation

Problematic Area	Potential Symptoms	Strategies	Potential Outcomes	
			Positive Outcome	Negative Outcome
Dispositional Challenges				
Low self-efficacy	<ul style="list-style-type: none"> • Student complains that they are not appreciated by their mentor. • Student vocalises that they will never be a teacher. • Student has a negative view of themselves as a teacher. 	<ul style="list-style-type: none"> • Review student's work life balance to ensure sufficient sleep, proper nutrition, regular exercise, and adequate financial resources. • Review feedback from mentor, identify and learn to celebrate success and learn from failures. 	Student has a positive image of themselves as a computing teacher.	Student continues to have low self-esteem issues regarding their self-perception of being a teacher or they leave the program.
Lack of ambition to be a computing teacher	<ul style="list-style-type: none"> • Student content with marginal performance as a teacher in the classroom. • Students demonstrates a superficial commitment to be a teacher, exhibited in the comment "I thought I would 	<ul style="list-style-type: none"> • Recap the concept of marginal gains in teaching to improve a student's performance as a teacher and improve learners' learning in the classroom. • Review why the student 	Student articulates a desire to be a computing teacher which is manifested in the classroom.	Student demonstrates marginal performance as a teacher in the classroom.

Problematic Area	Potential Symptoms	Strategies	Potential Outcomes	
			Positive Outcome	Negative Outcome
	<p>give teaching a go...”</p> <ul style="list-style-type: none"> • Student exhibits a lack of passion regarding teaching computing. 	<p>desired to undertake training as a computing teacher.</p> <ul style="list-style-type: none"> • Students writes a letter to themselves explaining why they want to be a computing teacher. • Have student take a career assessment exam to determine best career match. 		
<p>Negative attitude to being trained as a teacher</p>	<ul style="list-style-type: none"> • Poor time management, arriving late to school, leaving early • Not prepared to teach lessons, for example lesson plan and resources not prepared in sufficient time for mentor to comment on. • Ignore constructive feedback, advice, and 	<ul style="list-style-type: none"> • Review weekly teaching timetable to identify planning sessions • Ensure that lessons are planned in sufficient time for mentor to provide constructive feedback that can be acted upon. 	<p>Student demonstrates a positive attitude towards being trained as a teacher, by being organized in terms of planning lessons and developing resources.</p>	<p>Student continues to display a negative attitude being trained as a teacher that negatively impacts student learning, which may result in being counselled out of the program.</p>

Problematic Area	Potential Symptoms	Strategies	Potential Outcomes	
			Positive Outcome	Negative Outcome
	guidance from mentor.	<ul style="list-style-type: none"> • Conduct an intervention and developmental plan of action. 		
Lack of achievement against teachers' competence standards	<ul style="list-style-type: none"> • Student failing to achieve against the teachers' competence standards as reviewed by their mentor. 	<ul style="list-style-type: none"> • Review with student their progress to date against the teachers' competence standards and identify reasons for lack of progress. • Establish SMART targets for student to demonstrate achievement against teachers' competence standards. 	Student begins to make progress against the teachers' competence standards.	Student continues to make lack of progress against teachers' competence standards and placed on a support plan. Failure to make any progress may result in termination of their school experience.
Unwillingness to act upon advice and guidance from their mentor	<ul style="list-style-type: none"> • Student avoids meeting with their mentor. • SMART targets against teacher standards are repeated weekly. • Claims by student that mentor is not supportive, harsh towards 	<ul style="list-style-type: none"> • Tripartite reconciliation meeting between subject, mentor, and lecturer to discuss issue and identify way forward. • Student placed on negotiated 	Professional relationship between mentor and student restored, student now acting upon advice and guidance and making progress against the	Ultimately, a break down in professional relationship between mentor and student which could result in termination of school experience.

Problematic Area	Potential Symptoms	Strategies	Potential Outcomes	
			Positive Outcome	Negative Outcome
	them and even bullying them.	support plan with agreed SMART targets for student to achieve. ● Review support plan	teacher standards.	
● TPCK Challenges				
Poor computing content (subject) knowledge	<ul style="list-style-type: none"> ● Unable to contextualise a specific computing topic. ● Incorrect use and application of computing subject vocabulary. ● Unable to communicate computing concepts, principles, and subject matter. ● Unable to model a computing technique correctly. ● Unable to identify and address students' misconception(s) about a computing topic. 	<ul style="list-style-type: none"> ● Review and refresh student's computing subject knowledge audit. ● Review key vocabulary and associated definitions for a specific computing topic. ● Review knowledge organiser for specific computing topic to identify sequence of what needs to be taught, and how identified misconceptions are to be addressed. 	Student develops their subject knowledge of a specific computing topic and can confidently teach it clearly and coherently, and address students' misconceptions.	Student continues to demonstrate poor subject knowledge which may result in negatively impacting learners' learning. Ultimately, may result in termination of school experience.

Problematic Area	Potential Symptoms	Strategies	Potential Outcomes	
			Positive Outcome	Negative Outcome
		<ul style="list-style-type: none"> • Build a shared resources library • Devise SMART targets to address student's subject knowledge shortcoming. 		
Limited pedagogical knowledge for teaching computing	<ul style="list-style-type: none"> • Not using a range of teaching, learning and assessment strategies in teaching. • Ineffective modelling of a specific computing topic. • Lack sufficient application of systems thinking into instructional strategies. 	<ul style="list-style-type: none"> • Review the rationale for student using a limited range of teaching, learning and assessment strategies in a sequence of computing lessons. • Revisit the pedagogical approaches that were taught and modelled at university. • Identify pedagogical approaches for a student to adopt and demonstrate on school experience. 	Student demonstrates that they can effectively use a range of teaching, learning and assessment strategies in teaching a sequence of computing lessons.	Student is still reliant upon a limited range of teaching, learning and assessment strategies in teaching a sequence of computing lessons. Ultimately, may impact negatively upon learners' experience in the computing classroom and may result in termination of school experience.
Inappropriate	• Inappropriate modelling and	• Review with student the	Student demonstrates	Student continues to

Problematic Area	Potential Symptoms	Strategies	Potential Outcomes	
			Positive Outcome	Negative Outcome
technological knowledge for teaching computing.	<p>usage of technology for teaching, learning, and assessing a specific computing topic.</p> <ul style="list-style-type: none"> • Unable to explain and model clearly and concisely to learners how to use the technology to learn a specific computing topic. 	<p>rationale for their usage of technology.</p> <ul style="list-style-type: none"> • Critique how the student is using technology to teach a computing topic. • Model to student how to appropriately use technology to teach a computing topic. 	that they can effectively use and illustrate appropriate use of technology for teaching computing.	use technology either ineffectively or inappropriately which may impact negatively upon the learning of the learners in their computing class and may result in termination of school experience.
Lack of aptitude to be a teacher	<ul style="list-style-type: none"> • Unwilling to communicate in the classroom by either providing information or giving clear and concise instructions. • Unwilling to lead in the classroom, constantly referring to mentor. • Unwilling to transition into becoming a positive mentor to students. 	<ul style="list-style-type: none"> • Review taught sessions, video case analysis, on how to communicate in the classroom. • Review taught session on leading learning. • Use reflective portfolio • Classroom simulation 	Students communicate clearly and concisely information and instructions and takes initiative for leading in the classroom.	Students neither communicate effectively in the classroom, nor leads learning in the classroom. Negatively impacting the placement school from accepting future teaching candidates.

The previously identified complexities in teacher formation involve complex interpersonal aspects during the short period of metamorphoses from novice to confident, competent, capable, and creative computing teacher. Consequently, success during formation can be viewed from different perspectives. Celia must acknowledge and resolve the intrinsic and extrinsic challenges that they may face (Figure 2). Mastery of computing instruction (pedagogical knowledge), selection of appropriate computing subject knowledge instruction to a specific class (content knowledge) and use the most appropriate tools to teach computing (technological knowledge) will assist Celia to see the interaction and interconnection between pedagogical, content, and technological knowledge. Her professional practice could increase confidence, capability and competence in applying theoretical concepts to systems thinking in the classroom. During placement, Celia must establish a professional relationship with her school-based mentor. The mentor monitors her progress against the required teacher's competences and provides constructive feedback on improving her professional practice before she completes her placement. Other underlying obstacles may include Celia's ambition, attitude, and aptitude, insufficient achievement against teachers' competence standards, and willingness to act upon constructive feedback. The institution is vindicated in recognizing Celia's potential during her application process into the program, investing time and effort in educating Celia. Annual reports on negotiated key performance indicators, such as, number of applicants applying to the institution, number of candidates recruited against negotiated program target, number of successful pre-service teachers completing the program, and percentage of students gaining employment as teachers, are used for both programme and institutional continuous quality assurance and reported to numerous governmental and accreditation agencies.

6.2.7 Summary

Celia's journey through the different stages in each country to become a professional computing educator identified challenges from teacher formation to career placement. The work highlighted the importance of integrating TPCK and systems thinking into the realm of computing and STEAM education. This international case study used a global lens to compare Celia's journey in each country to become an effective and inclusive computing educator framed by systems thinking.

The main benefits for the researcher are:

- 1) an epistemological crafted set of problematic areas, the symptom signaling the difficulties, and a set of strategies to overcome these difficulties. This approach, as done in other computing fields, represents the initial seed of a databases of misconceptions and strategies to avoid them ()
- 2) A set of best practices in teaching computing crafted according to TPCK and system thinking approach

- 3) A set of resources supporting the daily teaching practices along with a set of reflective questions that can be used to frame the end of year analysis and next year planning
- 4) A multi country comparison of teaching practices and curricula

Allowing researchers and educators to have a research based set of best practices and guidelines to design, develop and assess a computing curricula for pre-service teachers formation and the professional development of the educators workforce.

Table 6 *Reflective Questions and Resources*

Reflective Questions	
How does your formation compare to Celia's?	Describe how you would increase your knowledge and competencies in computing education after employed as a computing educator?
Which country does your formation closely mirror and why?	What are your strongest attributes, aptitudes, and attitude that would make you an effective computing educator?
Select a computing education topic in STEAM, to describe how your lesson would integrate TPCK and a type of systems thinking?	Which areas do you need to improve to increase your effectiveness as a computing educator?
Explain why the mastery of systems thinking and TPCK would create a positive impact on P12 student learning.	Compile a list of negative attributes, aptitudes, and attitudes of your worst teachers. Do you have or present any of the negative attributes, aptitudes, and attitudes? If you do, how will you change those traits and behaviors?
How would you measure your effectiveness as a computing educator on your students' learning?	Why do you want to become a computing educator?

Resources	
Description	Website
Association of Computing Machinery (ACM)	https://www.acm.org
Computing Education Advocacy	https://code.org
Computer Science Teachers Association	https://www.csteachers.org
England Computing Education	https://royalsociety.org/topics-policy/projects/computing-education/
Institute of Electronic and Electrical Engineers Education Society	https://ieee-edusociety.org
International Society for Technology in Education	https://www.iste.org
K-12 Computer Science Framework	https://k12cs.org
National Centre for Computing Education	https://www.stem.org.uk/audience/secondary-computing
The TPACK Framework	http://tpack.org
The Systems Thinker	https://thesystemsthinker.com/introduction-to-systems-thinking/
European schoolnet	http://www.eun.org/
Scientix portal	http://www.scientix.eu/
Technological resource: Block based programming environments: App Inventor, Scratch, Snap! and its dialects	https://github.com/Code-WvS/awesome-snap
Technologies: low-cost computer Microcontroller	https://www.raspberrypi.org/ https://www.arduino.cc/
Content resources: open book repository	https://open.umn.edu/opentextbooks/textbooks?term=computer+science&commit=Go https://github.com/EbookFoundation/free-programming-books
Content resources: assessment resources – Quantum Project	https://diagnosticquestions.com/quantum https://community.computingatschool.org.uk/resources/4382/single
Pedagogical resources: peer instruction	http://peerinstruction4cs.org/
Pedagogical resources: POGIL	https://cspogil.org/Home

6.4 Professional work

Among the profession work performed inside this doctoral period it is possible to recall:

- 1) Co-leader of the ITICSE 25th anniversary committee

- 2) Convenor at the special Interest Group in Educational Technologies of the British Educational research
- 3) Member of the American Educational Research Association
- 4) Member of the Computer Science Teachers Associations committees:
 - (a) Computer Science Teachers Associations Professional Development Committee
 - (b) Computer Science Teachers associations 2023 Conference
- 5) Member of the 3rd Italian Scientix conference organizing committee
- 6) Member of the Computing at School
- 7) Guest editor for the Informatics in Education Journal
- 8) Special issue proposal for the IEEE Transaction on Education

6.3.1 ITiCSE 25th anniversary committee

The work as a co-chair of the committee was performed starting in December 2019. According to (Cassel, 2020) in December 2019, a group of volunteers started virtual meetings to discuss how to celebrate the 25th ITiCSE anniversary. Many initiatives were proposed and put forward but due to the pandemic situation and the possibility that the conference could be cancelled, a decision has been taken to postpone the celebration to 2021. However, the willingness to hold the conference was stronger than the crisis and the 25th conference took place in a virtual setting. As stated by the next organizer in the closing events, the "bar was set high" this year and the organizers were clever in turning a crisis into an advantage. The decision gave many the possibility to participate and to enjoy the conference in the virtual setting. The experience opens the door to exploring new organizational avenues for future SIGCSE conferences. The 25th celebration of the conference will take place at ITiCSE 2021 in Paderborn. The ITiCSE 25th anniversary committee will have ample time to deploy the proposed activities and to put forward new ones with the help of new volunteers for which an open call will be launched this September.

The ACM *Inroads* call for contributions⁹⁵ is one of the initiatives the committee put forward. The aim of the call was twofold: on one hand to celebrate the 25th anniversary of ITiCSE, and on the other hand to build a bridge between what has been done and what will be done. The call aims at giving a voice to teachers, educators, and researchers, welcoming contributions spanning:

- 1) history shared by the people who planned, organized, and participated in the 25 editions of the conference,
- 2) learning resources, teaching practices, and technologies covering all aspects of the Technological Pedagogical Content Knowledge Framework, and

⁹⁵ <https://dl.acm.org/action/showBmPdf?doi=10.1145%2F3399722>

- 3) methods to convey trends over the years arising from data demonstrating the evolution and uniqueness of the ITiCSE conference.

The ACM *Inroads* editors and the guest editors made every effort to spread the call starting from all the communication channels of the ACM SIGCSE: besides the ACM *Inroads*, the SIGCSE members listserv, and this *Bulletin*. The initiatives obtained the support of other associations starting from the Association for Teacher Education in Europe (ATEE) and teachers' Communities of Practices (CoP) like Computing At School and Scientix, with the goal of an even closer collaboration with teachers. Other collaborations have been sought with research associations in Europe, America, and all over the world.

Nevertheless the 2021 ITiCSe conference was an online-only conference. The aim of the commission was to organize event to be celebrated during a face to face conference so the celebration events never took place resulting in cancelling all the planned events

6.3.2 Convenor at the special Interest Group in Educational Technologies

The convenor experience spanned the 2020 and 2021. The activities performed, according to the official site⁹⁶, are summarized in the following figure

⁹⁶ <https://www.bera.ac.uk/person/francesco-maiorana>

Francesco Maiorana's contributions

Educational Technology SIG Annual Meeting

Join the Educational Technology SIG annual meeting to network with colleagues, learn about the activities of the SIG over the last year and plans for 2021. This meeting will be hosted on Zoom,...



Past event • 26 Mar 2021

Beyond 'Panic and Crisis Schooling': Aiming for 'Deep' Learning in Remote and Hybrid Environments

Online registration for this event has now closed, please email events@bera.ac.uk to register. #BERA_Technology The SDG agenda reflects growing recognition that human success and resilience...



Past event • 19 Mar 2021

Beyond 'Panic and Crisis Schooling': Aiming for 'Deep' Learning in Remote and Hybrid Environments

#BERA_Technology The SDG agenda reflects growing recognition that human success and resilience in the 21st century is no longer about accumulating knowledge, but about extrapolating from what we...



Video • 19 Mar 2021

Stay calm and e-educate: Technology, education and mental wellbeing

#BERA_MentalHealth In unprecedented times, maintaining education for learners across the spectrum has been a huge challenge, requiring educators to adapt quickly to online teaching and...



Past event • 18 Feb 2021

Educational Technology and Covid-19

This episode of the BERA Podcast explores the impact of Covid-19 on Educational Technology



Podcast • 26 Jul 2020

BERA news

BERA launches new Report on Education: The State of the Discipline

BERA in the news • 11 Jan 2023

2023 BJET Fellowship

News • 5 Dec 2022

2022 BERA Public Engagement and Impact Award Winners

News • 5 Dec 2022

BERA Race Equality Policy – 2022 Snapshot

News • 1 Dec 2022



[See all news](#) [Terms and Conditions](#) [Direct Debit Guarantee](#) [Cookie Policy](#)

British Educational Research Association 9–11 Endsleigh Gardens London WC1H 0EH

020 7612 6987 enquiries@bera.ac.uk

© BERA 2023 • Charity Number: 1150237 • Designed and developed by Soapbox

6.3.3 American Educational Research Association

The participation to the American Educational Research Association has brought the possibility to engage in research in other domains and collaborating to a distributed work on

related to the Palgrave Handbook of Educational Thinkers. The work (Cristaldi, under review) offers food for thought on Seneca, educator and philosopher, who lived in imperial Rome 2000 years ago, in an extremely complex historical context due to the multiple economic, political and cultural changes taking place at the time. This complexity, which in many respects recalls the many facets and problems of our own society, makes the thinking of the philosopher, who seeks to give answers to the most intimate needs of individuals, profoundly timely. Seneca believes in the educational role of philosophy, which has the task of bringing human person to wisdom, in a path that lasts a lifetime, and the role of an educator who supports the learner is fundamental.

6.3.4 Computer Science Teachers Associations committees

The activities with the Computer Science Teachers Association (CSTA) spanned the organization of the 2023 CSTA conferences where we just completed the selection of workshops, sessions, mini sessions and posters, to the three years involvement in the CSTA Professional Development committee started in 2022. Inside this group, besides reviewing all the Professional Development courses we are engaged in a research guided approach for reviewing the evaluation rubric linking the rubric to the Teacher and student standards.

6.3.5 3rd Italian Scientix conference organizing committee

The work of the commission is synthesized in the Book of abstract⁹⁷ with a particular emphasis on the introduction where, also in light of the submitted special issue for the IEEE Transaction on Education currently under review it is emphasized that Coding, computational, algorithmic, design, creative and critical thoughts (Caspersen, 2022) can be put at the service of all disciplines in a common effort to develop global skills (Mansilla, 2022) and contribute to the achievement of the OECD Council recommendations⁹⁸, to which the Council of Europe has contributed, to create better opportunities for young people. This Scientix conference has promoted a common effort in this direction by involving teachers, researchers, institutions and all stakeholders who support the Scientix community of practice. The committee is currently working to a book collecting along with research guided reflections on the educational practices, selected teachers' experience reports.

6.3.6 Computing at School

The work with the Computing at School communities of practices⁹⁹ has evolved across several years starting in 2012. Through the year the collaboration has evolved in collaboration inside the co-leded ITiCSE 2015 working group¹⁰⁰, collaboration with Project Quantum: tests

⁹⁷ https://www.indire.it/wp-content/uploads/2022/10/BoA_Scientix_Italia_2022_R.pdf

⁹⁸ <https://legalinstruments.oecd.org/en/instruments/OECD-LEGAL-0474>

⁹⁹ <https://www.computingatschool.org.uk/>

¹⁰⁰ <http://www.iticse2015.mii.vu.lt/en/menu1/presenters/working-groups/>

worth teaching to¹⁰¹, collaboration in preparation of two ITiCSE panel, and posting of events in the community of practices.

6.3.7 Guest editor for the Informatics in Education Journal

The work has guest editor started with the call for paper¹⁰² addressing the symbiotic relationship between industry and academia worldwide in establishing, evangelising, and embedding engineering education within a constantly changing computing curriculum. As reported in (Maiorana, F., & Csizmadia, A. (2022)) the special issue offers a variegated view of collaborations between academia and the commercial sector. The first group of papers deals with live educational experiences designed and developed with industries. In particular, CHANG and SHOKROLAH SHIRAZI present experiences related to capstone projects in software development. CONDORELLI and MALCHIODI reflect on the design and development of a master program in data science while VINCENTI presents a multiyear, multicourse collaboration experience with NASA as the industrial partner. This is in line with case studies utilizing open-source software (Ellis, 2015). An internship experience is explored by BUSUTTIL outlining the collaborative tripartite relationship between academia, industry, and the individual student. The second group of papers offers a set of tools, methods, and instruments useful in the interaction between education and enterprises. In particular, the importance of socio-affective relations both in presence and in an online setting is highlighted by AKAZAKI et al. who discuss pedagogical strategies useful in the daily teaching practice. TRIANTAFYLLOU and GEORGIADIS present a systematic literature review on gamification design patterns and their use in an industry educational setting. While TURAN-GÜNTEPE and ABDÜSSELAM present a tool for competencies determination in the realm of education 4.0 with a broad eye to industry 4.0 competencies. Finally, WILINSKI et al. present a method, based on multiple intelligence, for selecting and recruiting informatics students for the IT market. It is the editors' intention that this special issue, firstly celebrates the symbiotic relationship between education and industry as they collaborate to address the United Nations' sustainable developmental goals and ensure that all learners have equal opportunities as indicated by the OECD council recommendation. Secondly, it provides a challenge to all computing educators as to how they can ensure that the computing curriculum they teach is accessible, relevant, concurrent, and able to ensure the curiosity for a robust lifelong learning path contributing to reducing the number of NEETs (Maiorana, 2022).

¹⁰¹ <https://www.computingatschool.org.uk/resource-library/2016/april/project-quantum-tests-worth-teaching-to>

¹⁰² <https://infedu.vu.lt/journal/INFEDU/attachment/15>

6.3.8 Special issue proposal for the IEEE Transaction on Education

The special issue for the IEEE Transactions on Education entitled Coding, Computational, Algorithmic, Design, Creative, and Critical Thinking in K-16 education welcomes contributions from researchers, educators, high school teachers and their students, and undergraduate students with their professors, on experiences of schools and universities connected through research. By leveraging on previous special issues and editorial guidelines (Mitchell, 2020), (Mitchell, 2021), (Chance et al., 2019), (Damaj et al., 2020), this special issue intends to distill best practices of educational experiences in using coding and computing considered key competencies and essential way of thinking in all aspects of life: from algorithms to user interface design, from creative to critical thinking. This way of thinking has to be applied in multiple fields, such as computing, arts, and humanities, with a special attention and an integrated view (Tasiopoulou et al., 2020) to the disciplines of science, technology, engineering, and mathematics. The ultimate goal will be sharing best practices and guidelines on how create learning communities, from primary to higher education, to foster the development of global competencies able to guide citizens in their educational and working path thus reducing the percentage of young adult neither in employment nor in education and training (NEETs) (Maiorana et al., 2022b), (Maiorana & Cristaldi, n.d.).

Collaborative projects between schools and universities are also welcomed, perhaps offering bridges between undergraduate students acting as mentors for high school students and mutual mentorship opportunities between researchers, educators, parents, and students (Smith & Flores, 2019), (Destin et al., 2018), (Beninson et al., 2013), (Hadfield & Schweitzer, 2009), (Russell et al., 2007). The intent of this call is to seek best practices and to document the impact of actions such as tutoring, counseling, mentorship, near mentorship (Smith & Flores, 2019), and actions towards enhancing relationships which contribute to a more inclusive educational community. This could be applied both in formal settings, such as a tutor for the training of newly appointed teachers and teacher students (Durando et al., 2019), and in non-formal settings, such as communities of practices for outreach activities (Nistor et al., 2018).

The special issue aims to investigate one or more of the following research questions:

- 1) How to enhance learners' broad spectrum of 21st-century competencies (Clear et al., 2020b) and thinking skills, and how to apply these in civic engagement and real-world contexts.
- 2) How to connect the educational pipelines, from kindergarten to undergraduate diploma, involving all the actors of the formal, non-formal, and informal educational process emphasizing best practices, impacts, and sustainability of the activities.
- 3) How to enhance the relationships and dialogue between actors of the educational community: student-teacher, teacher-teacher, teacher-researchers, outreach communities, and all stakeholders involved.

6.5 Interlink of research and on the field educational practice

In the work (Maiorana, under review) after reviewing how to reach quality education and which competencies students should reach, presents the Scientix project. With a participatory action research and peer-led experiential learning approach, experience reports will be shared as a proof of concept about educational intervention coupling research and practices. The work highlight the importance of research method and approach in guiding all the phases of the educational practice: from course design, development and deployment to course assessment and self reflection both post intervention and longitudinal. In particular in (Maiorana, under review) in order to collect the experience reports one of the authors launched a call to the Scientix ambassador who volunteered to engage in a research-based reporting activity to improve their research and teaching practice and nurturing the seed of a self-directed small community of research engaged teachers. Building on the experience reported in (Maiorana et al., 2022; Maiorana et al., 2020b) the process followed, the following step inside the self-selected sample of experienced teachers in the Scientix community of practice:

- 1) Call for experiences: topics, and grade band
- 2) Asynchronous online negotiations on topics among the authors and decision on topics and index
- 3) Discussion on a reporting template according to research guidelines such as (McGill et al., 2018)
- 4) Online sharing of the experience reports which were peer reviewed
- 5) Crafting a list of reference for reflections and discussions
- 6) Drafting of the discussion and conclusion with its peer review

Considerable importance has been attributed to this last point 6 the activity of collaboration, cooperation and revision between teachers. We know from research that reviewing someone else's work can be a powerful learning mechanism. Furthermore, the exchange of good practices and the comparison of pedagogical ideas with peer colleagues is an effective way to evaluate and develop one's own practice and also recognize the different point of view. This has been demonstrated at all levels, from research to daily teaching practice especially in high school and tertiary education (Finkenstaedt-Quinn et al., 2021), (Bangert-Drowns et al., 2004). Writing activities are examples of STEAM collaboration and their positive effects goes far beyond the mere discipline to include Personal and Social Development (Anderson et al., 2015). Scalability to large class through the support of Artificial intelligence has been recently addressed (Davies et al., 2021). Leveraging on previous experiences nurtured also in the organization of a national conference for the whole community of practice (Niewint et al., 2022) where more than 60 experience reports from teachers spread in the whole national territory and covering all the grades in primary and secondary school for K1 to K13, thios work will summarize the didactic experience of a representative sample of the Scientix community of practice. The activities presented and the contents selected are fully

intellectually justified on the condition that they maintain a visual field that recognizes and conceives the existence of the interconnections of knowledge, skills and competencies with solidarity (Morin, 2000).

Chapter 7: Data related to Computer Science education

Objective of the chapter: envisage the use of attention metadata in respect to users, their queries and the type of attention, i.e., action performed on the resource

Approach: Propose a set of data to collect and a set of similarity metric useful for query recommendations.

Result achieved and novelty: an activity matrix collecting data in three dimensions: resources, users and queries.

Significance for the state of the art and the narrative of the work: attention metadata schema.

Learning resource repositories and assessment banks have been promoted in computing education fostering sharing, reuse and repurposing of learning assets. To grant users the best possible experience, respecting their privacy, we envisage the use of attention metadata in respects to users, their queries and the type of attention, i.e., action performed on the resources. The central idea is to use the query issued by the users (all fields used by the user can be taken into consideration with a particular emphasis on the terms and phrases used in the search). The consideration arises from the literature: every review, survey or meta-analysis paper begins by reporting the used search string. This reflects also the way users interact with databases, learning resource repositories, search engines and so on to perform queries to retrieve LR, scientific papers and so on. The envisaged use case is based on the following sequence of steps:

- 1) the user issues a query to the platform.
- 2) weighting and ranking of the retrieved LR using attention metadata
- 3) help user in choosing the relevant LR
- 4) update attention metadata
- 5) in line recommendations
- 6) off line recommendations

We will deal with the three major areas for ranking metrics, namely LR or topics, users and resources.

7.1 Topic Indexes

The indexes in this category should give a ranking of the LR based on the current user goal derived from the user queries. A vast amount of literature exists on topics ranking in related fields such as Information Retrieval that could be worth looking at. As example of this kind of indexes based on attention metadata information we recall:

Query topics index. This index could be a variation of the one proposed in (Ochoa, 2008) where it is based on the weighted sum, over the total number of queries, of the times the LR is considered in the query. The weighting scheme is based on the distance between the user

query and the query under observation. Variations of the index could use a different distance function as the one proposed in (Li, 2003). The set of queries can be restricted to the queries that have a similarity index with the issued query above a threshold (or in the same clustering class). Other considerations could regard a weighting of the different types of consideration that the LR could have received, how far in time the LR received the attention and include in the weighting scheme an index to evaluate the user that considered the LR (see user reliability indexes defined later).

User topics index. In a similar manner we can compute a weighted sum, over the total number of users, of the times the LR is considered by the user. In this case it could be convenient to restrict the user to those most similar to the current user (or in the same clustering class). The similarity could regard both the friend of a friend (FOAF) profile and the more dynamic attention profile that reflects recent user search patterns. Many similarity measures could be adopted. Variations, similar to the one presented for the previous index could be implemented.

7.2 User reliability indexes

The index should indicate the resource's owner reliability. The user reliability can be based on a combination of the user citation metrics and feedback given by other users both for resource itself and the resource's creator. These two factors simulate an automatic user evaluation based on indexes and an evaluation based on peer review summarizing the positive effects of the two approaches. Both these indexes based on impact factors and the average feedback can be weighted on the user index that made the action (repurpose, companion, download, leave feedback). In the long run the system can consider time by adopting mining algorithms suitable for time series data as reported (Yen, 2010).

User resources popularity indexes. This category of indexes is used to summarize with an index the number of times the user's LR have been used in some manner by someone else. For every action or for every category of action we obtain an index (or a set of indexes). We have highlighted two categories of actions: the citation that refers to more important (in terms of attention) actions of repurposing and companionship, and the consideration action that refers to all other actions signaling an interest by some user on the resource. The information on the two categories of actions are also conceptually separated and must be stored in two separate metadata schemes: the first in the LR schema, the second in the attention metadata scheme.

User Citation indexes. These indexes can be automatically computed by the platform on the basis of the number of times resources of the user have been repurposed or used in a companionship relation. These three actions, here referred to as citations, are given in order of importance and can be weighted to reflect this different importance. The set of resources created (uploaded) by the user can be restricted to the resource in the context of the query.

As suggested in (Derrick, 2011) several indexes can be used. It is also possible to adopt a combination of the proposed indexes:

- h-index. Hirsch's index depends on both the number of authors LR and the number of citations of these LR. It is defined as "a scientist has index h if h of his or her N_p papers have at least h citations each and the other $(N_p - h)$ papers have $\leq h$ citations each".
- m-index. It is the median number of citations received by LR that have a ranking $\leq h$
- m-quotient. Defined as the ratio between the h-index and the number of years (months) from the first LR publication.
- q2-index. It is the geometric mean of the h-index and the m-index (the square root of the product of the h- and m- indices). This index summarizes the quantitative dimension (number of LR) and the qualitative dimension (the impact of the LR).
- User Consideration indexes. These indexes, computed as the previous one, are based on the number of times some resource of the user has been considered. For consideration we mean, as stated before, any action that demonstrates an interest in the resources such as (selected, followed link, bookmarked, downloaded, rated (plus/minus), suggested to a friend and so on).

User feedback indexes. The indexes in this category should summarize and give a numeric representation of a sort of overall peer review on the owner of the resource. We can imagine two types of indexes (with a strong preference for the first one: rate the work not the person):

- Average user resources rating. This index is given by the average rating on all the user resources (eventually restricted to the resources in the context most similar to the initial query) left by other users- This index could be weighted with the reliability index of the user leaving the rating on the resource in order to incorporate a measure of her authority.
- Average owner rating. This index is obtained by averaging the rating obtained by the owner of the resource.

7.3 Resource Indexes

Resource popularity indexes. This category of index is used to summarize, in a similar way as the method used for users, the popularity of the single resource:

- Resource citation index. The index could be a variation of the citation index proposed in (Yen, 2010): a normalized weighted mean over time of the number of times the resource is "cited". The weight can also depend on the user index hence on the reliability of the user that performed the action on the object. This index gives a numeric representation of the number of times the resource has been repurposed or used in a companion relation. Eventually an index for each action could be reported.
- Resource usage index. This index could be a weighted mean of the number of actions performed on the resources. The actions can be selected from the considered one

(Selected, Followed link, Bookmarked, Downloaded, Rated with a distinction of positive and negative rates, Suggested to another user, ...) and weighted accordingly to the importance of the action. Eventually an index for each action could be reported.

Resource rating index. This index is the average of the explicit ratings left by other users on the resource. The ratings should be weighted on the user index to give more weighting to more reliable users.

7.4 Use of attention metadata

The attention metadata can be used in at least two phases: for ranking of the retrieved LR based only on the attention metadata; for recommendation

In each phase we can use attention metadata collected at each three levels to answer questions at each of the three levels.

We first propose metrics to recommend. This can be achieved by using an hybrid approach, that utilizes both techniques commonly employed in recommender systems:

- Collaborative Filtering techniques
- Content-based techniques.

The content-based techniques can make use of exploratory search infrastructures described in other contexts (Maiorana, 2015), (Maiorana, 2012). In the following, the assumption of using only attention metadata collected automatically by the platform. Thus, to recommend the objects, it is necessary to compute or infer the similarity between Resources, Queries and between Users. Each similarity can be computed as a combination of factors (mostly derived from AM): e.g. a similarity between users can be computed on the basis of the queries they issued to the system or on the basis of the LR they considered. The similarity among the three-level architecture must regard:

1) **Query similarity**

- a. Based on query content
- b. Based on LR returned by the query

2) **User similarity**

- a. Based on the queries performed
- b. Based on LRs considered (i.e., the history of the activity performed on LRs)
- c. The user similarity might also consider other users' profiles (not based on Attention Metadata) if available in the platforms.

3) **Resource similarity** (only used for ranking)

- a. Inferred based on the attention given to LRs in similar queries or inside a session
- b. based on clustering (already available from exploratory search, which is regarded as an "orthogonal" recommendation system).

8. Conclusions

In this research guided learning journey, several milestones can be identified, some of which have been highlighted in the chapters of this paper. By summarizing the work presented here, the main reflections and conclusion will be shared:

- 1) From the international comparison of the intended curricula, the importance of formal reporting has emerged. The work presented in this document can be understood as a proof of concept and advocacy for self-reflection and reporting through research.
- 2) From the international comparison of the enacted curricula, the importance of teacher flexibility has emerged. To cope with this need, coupled with the even greater flexibility required by students when designing learning resources, it is imperative to have great variety at all levels: content, learning paths, type of media used in delivering the learning resources, type of assessment, level of deepening, formative and summative assessment, technologies, and so on.
- 3) The underlying research based international comparison of intended and enacted curricula for each grade band can inspire, support, and sustain learning resource design, development, and assessment across adjacent grade bands, i.e., for this work, the first two years of high school, final years of high school, undergraduate studies in this work. Chapter four is intended as a proof of concepts in this direction.
- 4) The comparison of successive editions of computing curricula and the process used for this comparison can be applied in other contexts, e.g., in relation to the previous point.
- 5) The competencies building process (Clear, 2020) performed through the curriculum design process must guide learning resource design, development and assessment enacting the identified competencies building process.
- 6) The importance of a research based collaborative effort and the work presented in Chapter 6 demonstrates that teachers and educators are a key resource in the development of enacted curricula and offer a view of their role as curriculum designers and as guides and coaches for curriculum consumption. Resources and reduced workload should be granted to teachers and educators, allowing for a sharper impact as learning resource designers offering their contributions in fulfilling the needs identified as most important for quality learning resources (Falkner, 2019).
- 7) Data and data analysis should complement and support educators and whole learning communities in their daily activities.

The work aims to contribute with a grain of sand to the

As further steps we plan to map the curricula presented in Chapter 4 to international reference frameworks, such as the Informatics reference framework for Schools (Caspersen, 2022) while reflecting on Coding, Computational, Algorithmic, Design, Creative, and Critical

Thinking in K-16 education acting as guest editor for the IEEE Transaction on Education special issue¹⁰³.

In the long run, the ultimate goal of each and every educational action is to reduce the number of people neither in employment nor education and training. Besides providing inclusive access to quality education for all, actions must be taken to support students in their learning path after compulsory education. Leveraging previous international experiences in introducing computing in the USA (Morelli, 2014), where teachers received a stipend to attend a summer professional development course on computing and then taught the subject in the following school year, we advocate for financing, at the national level, school-led projects, assuring resources and grants for graduating students to attend the first three years of undergraduate education. The same students during the three years grant period will serve as near mentors for students attending school. In this way, the undergraduate students will develop their leadership role (Maiorana, 2022) and, instead of performing unrelated work, they can work on domains related to their learning path and contribute, under the supervision of the school teachers, to a quality learning path for the students aiming for high school graduation. We invite further discussion of this, and further ideas related to the theme, in Italian conferences, starting with the Scientix one, calling for a round table among educational research institutes, e.g., INDIRE, national evaluation institutes, e.g., INVALSI, ministry, and political representatives. Assessing the impact of some of the curricula design principles applied in designing the enacted curricula presented in Chapter 4, e.g., the flipped content (Maiorana, 2020), the interactive, inverted and spiral curriculum (Caspersen, 2022), along with pedagogical approaches (Caspersen, 2022), and technologies in setting up and implementing learning activities through adjacent grade levels could be an underlying research idea to be implemented and tested in one of the above envisaged school-led projects.

With an even broaden view of achieving a holistic development of the entire learning community an initial proposal for a special issue has been submitted and is reported in the following. The special issue, if accepted will provide a guide for an initial reflection reflections and research efforts and will provide a supportive infrastructure for merging different competencies and research domains ranging from computer science, psychology, philosophy, neuroscience and learning science. Support for research funding will be seaked if compatible with the policy of the hosting institution.

8.1 Computing education: from content, pedagogies, and technologies, to cognitive, psychological, and philosophical aspects

The entire educational system has recently faced numerous challenges. On one hand, recent crises such as pandemic have significantly undermined the United Nations' efforts towards

¹⁰³ <https://iee-edusociety.org/special-issue-coding-computational-algorithmic-design-creative-and-critical-thinking-k-16-education>

achieving quality education for all. Moreover, the persistence of ongoing wars worldwide, including Europe, has hindered many from accessing basic necessities, education included, according to Maslow's hierarchy of needs.

On the other hand, disruptive technological advancements such as large language models, machine learning and artificial intelligence are both challenging and revolutionizing the educational approaches.

Given these considerations, it is evident that education has to adopt a comprehensive approach that encompasses a wide range of competencies, with a spectrum as broad as possible ranging from domain-specific skills to pedagogical, technological, psychological, and philosophical aspects.

With these premises in mind, this Special Issue welcomes quantitative, qualitative, and mixed-method research papers. We also encourage extended reviews of substantive research studies and/or technical reports that have been previously published. Furthermore, *Education Sciences* invites commentaries and original opinion pieces and/or analyses of issues related to the following topics:

- A) Computer Science Pedagogy (S. A. Fincher & Robins, 2019), (Hello world, 2021)
 - a. Peer Instruction & peerinstruction4CS
 - b. Parson problems
 - c. Process Oriented Guided Inquiry learning
- B) Disci, multi, inter and transdisciplinary computing Curricula Development (Maiorana, 2021c), (Maiorana, 2019a), (Taylor et al., 2018)
 - a. Artificial Intelligence and data science (Müller, n.d.), (Ozmen Garibay et al., 2023), (Decker & McGill, 2019)
 - b. Computer security
 - c. Accessibility
 - d. Digital and green transformations through the lenses of computing education
 - e. Civic education (Naval et al., 2022), (Cristaldi et al., 2022)
- C) Character and holistic development
 - a. Holistic student development (Datnow et al., 2022a)
 - b. Character education (Watts & Kristjánsson, 2022), (Vaccarezza et al., 2023)
 - c. Applied behavioral science with a special emphasis on Behavior change frameworks such as the Mindspace framework (Dolan et al., 2012)
- D) Special and gifted education needs with a special emphasis on Computer Science education
 - a. Universal Design for learning
 - b. AccessCSforAll (Ladner & Stefik, 2017)
 - c. Universal Design Learning for Computer Science (UDL4CS) (Twarek et al., 2021)
 - d. Supporting metacognition (Mitsea et al., 2023)
- E) From learning science to Science of Learning (Saleh & Khine, 2023)
 - a. Cognitive aspects of reasoning and problem solving (Fedorenko et al., 2019)
 - b. Cognitive psychology
 - c. Cognitive neuroscience (Scherer et al., 2019)

Learning theories in relation to computing education (Ko, Draper, et al., 2023), (McGill et al., 2022)

References for the thesis

- Abel, P. (2000). *IBM PC Assembly language and programming*. Prentice Hall PTR.
- A.C.A.R.A. (2016). *Australian Curriculum, Assessment and Reporting Authority*.
<https://www.australiancurriculum.edu.au/umbraco/Surface/Download/Pdf?subject=Digital%20Technologies&type=F10>
- ACM Code 2018 task force. (2018). *Using the code*. <https://ethics.acm.org/code-of-ethics/using-the-code/>
- ACM/IEEE. (2013). *ACM/IEEE Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. ACM/IEEE.
https://www.acm.org/binaries/content/assets/education/cs2013_web_final.pdf.
- ACM/IEEE. (2017). *Information Technology Curricula 2017. IT2017. Curriculum Guidelines for Baccalaureate Degree Programs in Information Technology*. ACM/IEEE.
- ADLCS. (2018). *Alabama State Department of Education. Digital literacy and computer science: Course of study*.
- Ainley, J., & Carstens, R. (2018). *Teaching and learning international survey (TALIS) 2018 conceptual framework*.
- Alabama State Department of Education. (2018a). *Alabama Digital Literacy and Computer Science Standards Course of Study (K-12)*.
- Alabama State Department of Education, A. S. D. (2018b). *Digital literacy and computer science: Course of study*.
- Almstrum, V. L., Guzdial, M., Hazzan, O., & Petre, M. (2005). Challenges to computer science education research. *Proceedings of the Thirty-Sixth SIGCSE Technical Symposium on Computer Science Education, SIGCSE 2005:191-192*.
<https://doi.org/10.1145/1047124.1047415>
- Amoussou, G. A., Boylan, M., & Peckham, J. (2010). Interdisciplinary computing education for the challenges of the future. *SIGCSE'10 - Proceedings of the 41st ACM Technical*

- Apolone, G., & Mosconi, P. (1998). The Italian SF-36 Health Survey: Translation, validation and norming. *Journal of Clinical Epidemiology*, 51(11), 1025–1036.
- Arefin, A. S., Halim, S., Rahman, M. L., By, F., Revilla, M. A., & Prokashoni, G. (2006). *Art of programming contest: C programming tutorials, data structures, algorithms*.
<http://acmicpc-live-archive.uva.es><http://online-judge.uva.es>.
- Argyropoulos, V., Paveli, A., & Nikolarazi, M. (2019). The role of DAISY digital talking books in the education of individuals with blindness: A pilot study. *Educ Inf Technol*, 24(1), 693–709.
- Arthur, J., & Kristjánsson, K. (2022). *The Jubilee Centre framework for character education in schools*. Jubilee Centre for Character and Virtues. <https://www.jubileecentre.ac.uk>
- Atzeni, P., Ceri, S., Paraboschi, S., & Torlone, R. (1999). *Database systems—concepts, languages and architectures*.
- Bandura, A. (1997). *Self-efficacy: The exercise of control*. Macmillan.
- Bandura, A. (2006). Guide for constructing self-efficacy scales. *Self-Efficacy Beliefs of Adolescents*, 5(1), 307–337.
- Bandura, A., & Wessels, S. (1994). *Self-efficacy* (Vol. 4). na.
- Barba, L. A., Barker, L. J., Blank, D. S., Brown, J., Downey, A. B., George, T., Heagy, L. J., Mandli, K. T., Moore, J. K., & Lippert, D. (2019). Teaching and learning with Jupyter. *Recuperado: <https://Jupyter4edu.Github.Io/Jupyter-Edu-Book>*.
- Barendsen, E., Grgurina, N., & Tolboom, J. (2016). A new informatics curriculum for secondary education in the Netherlands. *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives*, 105–117.

- Barendsen, E., Mannila, L., Demo, B., Grgurina, N., Izu, C., Mirolo, C., Sentance, S., Settle, A., & Stupurienė, G. (2015). Concepts in K-9 computer science education. In *Proceedings of the 2015 ITiCSE on working group reports* (pp. 85–116).
- Bau, D., Gray, J., Kelleher, C., Sheldon, J., & Turbak, F. (2017). Learnable programming: Blocks and beyond. *Commun ACM*, 60(6), 72–80.
- Bauman, Z., & Leoncini, T. (2018). *Born liquid*. John Wiley & Sons.
- Bell, T., Alexander, J., Freeman, I., & Grimley, M. (2009). Computer science unplugged: School students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology*, 13(1), 20–29.
- Bell, T., Rosamond, F., & Casey, N. (2012a). Computer science unplugged and related projects in math and computer science popularization. *The Multivariate Algorithmic Revolution and beyond: Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday*, 398–456.
- Bell, T., Rosamond, F., & Casey, N. (2012b). Computer science unplugged and related projects in math and computer science popularization. In *The multivariate algorithmic revolution and beyond* (pp. 398–456). Springer.
- Bell, T., Witten, I., & Fellows, M. (2015). *CS Unplugged: An enrichment and extension programme for primary-aged students*.
- Bellettini, C., Lonati, V., Malchiodi, D., Monga, M., Morpurgo, A., Torelli, M., & Zecca, L. (2014). Informatics education in Italian secondary schools. *ACM Transactions on Computing Education (TOCE)*, 14(2), 1–6.
- Bellettini, C., Lonati, V., Monga, M., & Morpurgo, A. (2019). An Analysis of the Performance of Italian Schools in Bebras and in the National Student Assessment INVALSI. *TACKLE@ EC-TEL*.
- Bellettini, C., & Palazzolo, M. (2020). *Situated Learning with Bebras Tasklets*.
<http://aladdin.di.unimi.it>.

- Benaya, T., Zur, E., Dagiene, V., & Stupuriene, G. (2017). Computer Science High School Curriculum in Israel and Lithuania—Comparison and Teachers' Views. *Balt J Mod Comput, 5*(2).
- Beninson, L. A., Koski, J., Villa, E., Faram, R., & O'Connor, S. E., VI. (2013). Evaluation of the Research Experiences for Undergraduates (REU) sites program. *The Emerging Role of Exosomes in Stress Physiology, 118*.
- Bergin, S. (2006). A computational model to predict programming performance. *Maynooth University PhD Thesis*.
- Berland, M., & Lee, V. R. (2011). Collaborative strategic board games as a site for distributed computational thinking. *Int J Game-Based Learn, 1*(2), 65–81.
- Berry, M., Woollard, J., Hughes, P., Chippendal, J., Ross, Z., & resources, W. J. B. (2015). *Barefoot computing resources*.
- Billett, S. (2009). Realising the educational worth of integrating work experiences in higher education. *Studies in Higher Education, 34*(7), 827–843.
- Billon, N., Nistor, A., Mihai, G., Grizelj, A., Myrtsiote, E., Miklasinska, O., Pocze, B., & Gras-Velazquez, A. (2019). *Scientix Update—September 2019*. European Schoolnet.
- Binkley, M., Erstad, O., & Herman, J. (n.d.). Defining twenty-first century skills. In *Assessment and Teaching of 21st Century Skills* (Vol. 2014, pp. 17–66). Springer. https://doi.org/10.1007/978-94-007-2324-5_2
- Bishop, J. L., & Verleger, M. A. (2013). The flipped classroom: A survey of the research. *ASEE National Conference Proceedings, Atlanta, GA, 30*(9), 1–18.
- Bishop, J., & Verleger, M. (2013). The flipped classroom: A survey of the research. *ASEE National Conference Proceedings, 30*, 1–18.
- Blumenfeld, P. C., Soloway, E., Marx, R. W., Krajcik, J. S., Guzdial, M., & Palincsar, A. (1991). Motivating project-based learning: Sustaining the doing, supporting the learning. *Educational Psychologist, 26*(3–4), 369–398.

- Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing Computational Thinking in Compulsory Education* (Vol. 2016, pp. 1–68).
<https://doi.org/10.2791/792158>
- Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016). Developing computational thinking: Approaches and orientations in K-12 education. *EdMedia+ Innovate Learning*, 13–18.
- Bocconi, S., Chiocciariello, A., & Earp, J. (2018). The Nordic approach to introducing Computational Thinking and programming in compulsory education. *Report Prepared for the Nordic@ BETT2018 Steering Group*. [https://doi.](https://doi.org/)
- Bogliolo, A. (2016). Coding in your classroom, now! *Giunti Editore*.
<https://ora.uniurb.it/handle/11576/2642776>.
- Booch, G. (1991). *Object-Oriented Analysis and Design with Applications California Benjamin Cummings*.
- Borges, J. C., Oranges Cezarino, L., & Ferreira, T. C. (2017). Student organizations and Communities of Practice: Actions for the 2030 Agenda for Sustainable Development. *Elsevier*. <https://doi.org/10.1016/j.ijme.2017.02.011>.
- Borgonovi, F. (2015). Do teacher-student relations affect students' well-being at school. *PISA in Focus*.
- Bowers, D. S., & Howson, O. (2019). *Analysis of accreditation approaches in the Computing sector*.
- Brennan, K., & Resnick, M. (2012a). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the 2012 Annual Meeting of the American Educational Research Association*, 1, 25.
- Brennan, K., & Resnick, M. (2012b). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the 2012 Annual Meeting of the American Educational Research Association, Vancouver, Canada*, 1, 25.

- Britain), R. S. (Great. (2012). *Shut down or restart?: The way forward for computing in UK schools*. Royal Society.
- Brown, N. C. C., A, A., S, S., & Kölling, M. B. (2018). Five Years On: An Evaluation of a Large-scale Programming Data Collection Project. *Proceedings of the 2018 ACM Conference on International Computing Education Research (ICER '18)*.
<https://doi.org/10.1145/3230977.3230991>
- Bruce, K. B. (2018). Five big open questions in computing education. *ACM Inroads*, 9(4), 77–80. <https://doi.org/10.1145/3230697>
- Bruner, J. (1996a). *The culture of education*. Harvard University Press.
- Bruner, J. (1996b). *The culture of education*. Harvard University Press.
- Buck, J. W., & Perugini, S. (2019). An interactive, graphical simulator for teaching operating systems. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 1290–1290.
- Burgstahler, S. (2009a). Universal Design of Instruction (UDI): Definition, Principles, Guidelines, and Examples. *DO-IT*.
- Burgstahler, S. (2009b). Universal Design of Instruction (UDI): Definition, Principles, Guidelines, and Examples. *DO-IT*.
- Burgstahler, S. E., & Cory, R. C. (2010a). *Universal Design in Higher Education: From Principles to Practice*. Harvard Education Press.
- Burgstahler, S. E., & Cory, R. C. (2010b). *Universal design in higher education: From principles to practice*. Harvard Education Press.
- Burguillo, J. C. (2010). Using game theory and competition-based learning to stimulate student motivation and performance. *Computers & Education*, 2010(55), 566–575.
- Caena, F., & Redecker, C. (2019). Aligning teacher competence frameworks to 21st century challenges: The case for the European Digital Competence Framework for Educators (DIGCOMPEDU. *EUROPEAN JOURNAL OF EDUCATION, ISSN 0141-8211 (Online)*, 54(3), 356-369,.

- Cañas, A. J., Hill, G., & Carff, R. (2004). *CmapTools: A knowledge modeling and sharing environment*.
- Care, E., Griffin, P., & McGaw, B. (2012). *Assessment and Teaching of 21st Century Skills*. Springer.
- Carlisle, M. C. (2009). Raptor: A visual programming environment for teaching object-oriented programming. *Journal of Computing Sciences in Colleges*, 24(4), 275–281.
- Carpenter, L. B. (2015). *Assistive technology: Access for all students* (3rd ed.). Los Angeles, Pearson.
- Caspersen, M., Diethelm, I., Gal-Ezer, J., McGettrick, A., Nardelli, E., Passey, D., Rován, B., & Webb, M. (2022a). *Designing and Implementing a Concrete Informatics Curriculum for School*.
- Caspersen, M., Diethelm, I., Gal-Ezer, J., McGettrick, A., Nardelli, E., Passey, D., Rován, B., & Webb, M. (2022b). *Informatics reference Framework for School*.
- Caspersen, M. E., Diethelm, I., Gal-Ezer, J., McGettrick, A., Nardelli, E., Passey, D., Rován, B., & Webb, M. (2022). *Informatics Reference Framework for School*.
- Caspersen, M. E., Gal-Ezer, J., McGettrick, A., & Nardelli, E. (2018a). *Informatics for all the strategy*. ACM.
- Caspersen, M. E., Gal-Ezer, J., McGettrick, A., & Nardelli, E. (2018b). *Informatics for all the strategy*. ACM.
- Cassel, L., MacKellar, B., Peckham, J., Spradling, C., Reichgelt, H., Westbrook, S., & Wolz, U. (2014). Interdisciplinary computing in many forms. *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, 623–624.
- Celko, J. (2006). *Joe Celko's SQL Puzzles and Answers*. Elsevier.
- Celko, J. (2010). *Joe Celko's SQL for smarties: Advanced SQL programming*. Elsevier.
- Ceriani, M., & Bottoni, P. (2017). SparqlBlocks: Using blocks to design structured linked data queries. *Language (XSD, 1)*, 11.

- Cetin, I., & Ozden, M. Y. (2015). Development of computer programming attitude scale for university students. *Computer Applications in Engineering Education*, 23(5), 667–672.
- Chan, T.-W., Roschelle, J., & Hsi, S. (2006). One-to-one technology-enhanced learning: An opportunity for global research collaboration. *Res Pract Technol Enhanc Learn*, 1(01), 3–29.
- Chance, S. M., Williams, B., Goldfinch, T., Adams, R. S., & Fleming, L. N. (2019). Guest Editorial Special Issue on Using Enquiry-and Design-Based Learning to Spur Epistemological and Identity Development of Engineering Students. *IEEE Transactions on Education*, 62(3), 157164,.
- Chien, A. A. (2019). *Cracks in open collaboration in universities*.
- Chiquito, M., Castedo, R., Santos, A. P., López, L. M., & Alarcón, C. (2020). Flipped classroom in engineering: The influence of gender. *Comput Appl Eng Educ*, 28(1), 80–89. <https://doi.org/10.1002/cae.22176>
- Churchill, E. F., Bowser, A., & Preece, J. (2013). Teaching and learning human-computer interaction: Past, present, and future. *Interactions*, 20(2), 44–53.
- CINI. Consorzio Interuniversitario nazionale per l'informatica. (2017). *Proposal for a national Informatics curriculum in the Italian school*. National Interuniversity consortium for Informatics. CINI.
- Ciociola, C., & Reggi, L. (2015). A Scuola di OpenCoesione: From open data to civic engagement. *Open Data as Open Educational Resources*, 26.
- Clear, A., Clear, T., Takada, S., & Cuadros-Vargas, E. (2022). Comparing Global Curricula and Local Computing Degree programs using the CC2020 Curriculum Visualization Tool. *2022 IEEE Frontiers in Education Conference (FIE)*, 1–4.
- Clear, A., Clear, T., Vichare, A., Charles, T., Frezza, S., Gutica, M., Lunt, B., Maiorana, F., Pears, A., & Pitt, F. (2020a). Designing computer science competency statements: A process and curriculum model for the 21st century. In *Proceedings of the Working*

- Group Reports on Innovation and Technology in Computer Science Education* (pp. 211–246).
- Clear, A., Clear, T., Vichare, A., Charles, T., Frezza, S., Gutica, M., Lunt, B., Maiorana, F., Pears, A., & Pitt, F. (2020b). Designing computer science competency statements: A process and curriculum model for the 21st century. *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education*, 211–246.
- Codd, E. F. (1990). *The relational model for database management: Version 2*. Addison-Wesley Longman Publishing Co., Inc.
- Coddington, M. (2015). Clarifying Journalism’s Quantitative Turn: A typology for evaluating data journalism, computational journalism, and computer-assisted reporting. *Digit Journal*, 3(3), 331–348. <https://doi.org/10.1080/21670811.2014.976400>
- Code.org. (2018). *State of computer science education*. Code.org. Retrieved from https://code.org/files/2018_state_of_cs.pdf
- Codeorg. (2018). *State of computer science education*. Code.org. https://code.org/files/2018_state_of_cs.pdf
- Code.org. (2019). *State of computer science education. Equity and diversity*. <https://advocacy.code.org/20>
- Connolly, R. (2020). Why computing belongs within the social sciences. *Communications of the ACM*, 63(8), 54–59.
- Cook, D. (2015). Flowgorithm: Principles for Teaching Introductory Programming Using Flowcharts. *Proc. American Society of Engineering Education Pacific Southwest Conf.(ASEE/ PSW)*, 158–167.
- Coopersmith, S. (1967). The antecedents of self-esteem San Francisco. *H Freeman and Company*.

- Corradini, I., & Nardelli, E. (2021). Promoting digital awareness at school: A three-year investigation in primary and secondary school teachers. *Proceedings of EDULEARN21 Conference, 5*, 6th.
- Corral, L., & Fronza, I. (n.d.). *A Strategy for Assessing the Acquisition of Computational Thinking Competences: A Software Engineering Approach*.
<http://appinventor.mit.edu/>.
- Costantino, T. (2018). STEAM by another name: Transdisciplinary practice in art and design education. *Arts Education Policy Review, 119*(2), 100–106.
- Coughlan, T. (2020). The use of open data as a material for learning. *Educational Technology Research and Development, 68*(1), 383–411.
- CPUlator. (n.d.). *CPUlator Simulator Documentation*. <https://cpulator.01xz.net/doc/>
- Craig, M., & EngageCSEdu, M. B. (2020). How should we assess assignments? *ACM Inroads, 11*(1), 17–19. <https://doi.org/10.1145/3381024>
- Crick, T., & Sentance, S. (2011). Computing at school: Stimulating computing education in the UK. *Proceedings of the 11th Koli Calling International Conference on Computing Education Research, 122–123*.
- Cristaldi, G., Quille, K., Csizmadia, A. P., Riedesel, C., Richards, G. M., & Maiorana, F. (2022). The intervention, intersection and impact of social sciences theories upon computing education. *2022 IEEE Global Engineering Education Conference (EDUCON)*, 1561–1570.
- Cristaldi, G., Richards, G., Csizmadia, A., Riedesel, C., & Maiorana, F. (2020). SPECIAL EDUCATION FOR SPECIAL STUDENTS: EXPERIENCE REPORTS FROM TEACHERS' EDUCATORS AND IN THE FIELD TEACHERS. *ICERI2020 Proceedings*, 7601–7606.
- CSForALL. (2019). *Computer Science for ALL Students (CSForALL). 2019. Common Data Collection Survey and Support Document. (2019)*.

https://drive.google.com/file/d/1gi8lLlAbfpAvcffWn__hvBiRUQKEEMjF/view ;

<https://drive.google.com/file/d/1AqfPXw2ZyQ0sluDbxM-zK8Z0AQ4SYvu-/view>

Csizmadia, A., Curzon, P., & Dorling, M. (2015). *Computational thinking-A guide for teachers*.

Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T., Selby, C., & Woollard, J. (2015). *Computational thinking-A guide for teachers*.

Csizmadia, A., & Maidens, D. (2018). *Unlocking the Secrets of the Binary Box. London Computing Education Research Symposium*.

Csizmadia, A., Standl, B., & Waite, J. (2019a). Integrating the constructionist learning theory with computational thinking classroom activities. *Informatics in Education, 18*(1), 41–67.

Csizmadia, A., Standl, B., & Waite, J. (2019b). Integrating the constructionist learning theory with computational thinking classroom activities. *Informatics in Education, 18*(1), 41–67.

Csizmadia, A., Standl, B., & Waite, J. (2019c). Integrating the Constructionist Learning Theory with Computational Thinking Classroom Activities. *Informatics in Education, 18*(1), 4167.

CSTA. (2016). *K-12 computer science framework*. ACM.

C.S.T.A. (2016). *K-12 computer science framework*. ACM.

CSTA. (2017a). Computer Science Teachers Association. CSTA K-12 Computer Science Standards. *Revised*. <http://www.csteachers.org/standards>

CSTA. (2017b). *CSTA K-12 Computer Science Standards*. Computer Science Teachers Association. <http://www.csteachers.org/standards>

CSTA. (2017c). *CSTA K-12 Computer Science Standards*. Computer Science Teachers Association. <http://www.csteachers.org/standards>

CSTA. (2017d). *K-12 Computer Science Standards Revised 2017*. Computer Science Teachers Association. <https://csteachers.org/page/standards>

- CSTA. (2020a). *Computer Science Teachers Association. Standards for Computer Science Teachers*. <https://csteachers.org/teacherstandards>
- CSTA. (2020b). *Standards for Computer Science Teachers*. Computer Science Teachers Association. <https://csteachers.org/teacherstandards>.
- CSTA. (2020c). *Standards for Computer Science Teachers*. Computer Science Teachers Association. <https://csteachers.org/teacherstandards>
- Cutts, Q., Patitsas, E., Cole, E., Donaldson, P., Alshaigy, B., Gutica, M., Hellas, A., Larrazamendiluze, E., McCartney, R., & Riedesel, C. (2018). Early developmental activities and computing proficiency. *Proceedings of the 2017 ITiCSE Conference on Working Group Reports*, 140–157.
- Dagiene, V., & Futschek, G. (2008). Bebras International Contest on Informatics and Computer Literacy. *Criteria for Good Tasks In International Conference on Informatics in Secondary Schools-Evolution and Perspectives, 5090*, 19–30. https://doi.org/10.1007/978-3-540-69924-8_2
- Dagienė, V., & Stupurienė, G. (2016). Bebras-a Sustainable Community Building Model for the Concept Based Learning of Informatics and Computational Thinking. *Informatics Educ*, 15(1), 25–44. <https://doi.org/10.15388/infedu.2016.02>
- Damaj, I., El-Abd, M., A.Kranov, A., DeBoer, J., & Project-Based, J. G. E. S. I. (2020). Senior Design, and Capstone Courses in Engineering Education. *IEEE Transactions on Education*, 63(2), 79-81,.
- D’Anca, J.-A. (2017). *Mindset and resilience: An analysis and intervention for school administrators*. St. John’s University (New York), School of Education and Human Services.
- Datnow, A., Park, V., Peurach, D. J., & Spillane, J. P. (2022a). RESEARCH FOUNDATIONS. Transforming Education for Holistic Student Development: Learning from Education System (Re) Building around the World. Technical report. *Center for Universal Education at The Brookings Institution*.

- Datnow, A., Park, V., Peurach, D. J., & Spillane, J. P. (2022b). Transforming Education for Holistic Student Development: Learning from Education System (Re) Building around the World. Report. *Center for Universal Education at The Brookings Institution*.
- De Rosa, R., Ferrari, C., & Kerr, R. (2017). The EMMA experience. Emerging patterns and factors for success. *Digital Education: Out to the World and Back to the Campus: 5th European MOOCs Stakeholders Summit, EMOOCs 2017, Madrid, Spain, May 22-26, 2017, Proceedings 5*, 20–28.
- De Rossi, M., & Angeli, C. (2018a). Teacher education for effective technology integration. *Italian Journal of Educational Technology*, 26(1), 3–6.
- De Rossi, M., & Angeli, C. (2018b). Teacher education for effective technology integration. *Italian Journal of Educational Technology*, 1(26), 3–6.
- De Rossi, M., & Trevisan, O. (2018a). Technological Pedagogical Content Knowledge in the literature: How TPCK is defined and implemented in initial teacher education. *Italian Journal of Educational Technology*, 26(1), 7–23.
- De Rossi, M., & Trevisan, O. (2018b). Technological Pedagogical Content Knowledge in the literature: How TPCK is defined and implemented in initial teacher education. *Italian Journal of Educational Technology*, 26(1), 7–23.
- Decker, A., & McGill, M. M. (2019). A systematic review exploring the differences in reported data for pre-college educational activities for computer science, engineering, and other STEM disciplines. *Education Sciences*, 9(2), 69.
- Deitrick, E., & Stowell, J. (2019). 2019 survey of Computer Science teaching: Challenges and resources in U.S. *Colleges and Universities*. www.codio.com/research/2019-computer-science-teaching-survey
- Deng, W., Pi, Z., Lei, W., Zhou, Q., & Zhang, W. (2020). Pencil Code improves learners' computational thinking and computer learning attitude. *Comput Appl Eng Educ*, 28(1), 90–104. <https://doi.org/10.1002/cae.22177>

- Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Commun ACM*, 60(6), 33–39. <https://doi.org/10.1145/2998438>
- Deshpande, A. A., & Huang, S. H. (2011). Simulation games in engineering education: A state-of-the-art review. *Computer Applications in Engineering Education*, 19(3), 399–410.
- Destin, M., Castillo, C., & Meissner, L. (2018). A field experiment demonstrates near peer mentorship as an effective support for student persistence. *Basic and Applied Social Psychology*, 40(5), 269-278,.
- DFE, U. (2013a). *National curriculum in England: Computing programmes of study* (Vol. 16).
- DFE, U. (2013b). National curriculum in England: Computing programmes of study. Retrieved July, 16, 2014.
- DFE, U. (2013c). National curriculum in England: Computing programmes of study. Retrieved July, 16, 2014.
- Diagram Center. (2015). *Image description guidelines*. Retrieved [2019-11-30] from <http://diagramcenter.org/making-images>
- Diagram center. (2015). *Poet Image Description Tool*.
- Dix, A., Finlay, J., Abowd, G. D., & Beale, R. (2003). *Human-computer interaction*. Pearson Education.
- Dlab, M., Boticki, I., Hoic-Bozic, N., & CL-C&, E. (2020). Exploring group interactions in synchronous mobile computer-supported learning activities. *Elsevier*. <https://www.sciencedirect.com/science/article/pii/S036013151930288X>.
- Do-I.T. (2017). *DO-IT Retrospective Our First 25 Years*. University of Washington.
- Dolan, P., Hallsworth, M., Halpern, D., King, D., Metcalfe, R., & Vlaev, I. (2012). Influencing behavior: The mindspace way. *Journal of Economic Psychology*, 33(1), 264–277.

- Dolgopolovas, V., Jevsikova, T., & Dagiene, V. (2018). From Android games to coding in C. An approach to motivate novice engineering students to learn programming: A case study. *Comput Appl Eng Educ*, 26(1), 75–90. <https://doi.org/10.1002/cae.21862>
- Dorling, M., & Walker, M. (2014). Computing progression pathways. *Computing at School*.
- Dri, M., Antonopoulos, I. S., Canfora, P., & Gaudillat, P. (2015). *Best environmental management practice for the food and beverage manufacturing sector*. Final Draft.
- DuFour, R., & DuFour, R. (2013). *Learning by doing: A handbook for Professional Learning Communities at Work TM*. Solution Tree Press.
- Durando, M., Sjøberg, S., Gras-Velazquez, A., Leontaraki, I., Santolaya, E. M., & Tasiopoulou, E. (2019). *Teacher Training and IBSE Practice in Europe—A European Schoolnet overview*. European Schoolnet.
- Dweck, C. S. (2006). *Mindset: The new psychology of success*. Random House.
- Education ambivalence. (2010). *Nature*, 465, (7298), 525–526.
- Education ambivalence. (2010). *Nature*, 465(7298), 525–526.
<https://doi.org/10.1038/465525b>
- Education, E. C. H. L. G. on S., Science, E. C., & Economy. (2007). *Science education now: A renewed pedagogy for the future of Europe* (Vol. 22845). Office for Official Publications of the European Communities.
- Ellis, H. J., Hislop, G. W., Jackson, S., & Postner, L. (2015a). Team project experiences in humanitarian free and open source software (HFOSS). *ACM Transactions on Computing Education (TOCE)*, 15(4), 1–23.
- Ellis, H. J., Hislop, G. W., Jackson, S., & Postner, L. (2015b). Team project experiences in humanitarian free and open source software (HFOSS). *ACM Transactions on Computing Education (TOCE)*, 15(4), 1–23.
- Ellis, H., Morelli, R. A., & Hislop, G. W. (2008). Support for educating software engineers through humanitarian open source projects. *2008 21st IEEE-CS Conference on Software Engineering Education and Training Workshop*, 1–4.

- Elmasri, R., & Navathe, S. B. (2011). *Database systems* (Vol. 9). Pearson Education Boston, MA.
- Eraut, M. (2010). Knowledge, working practices, and learning. In *Learning through practice* (pp. 37–58). Springer.
- European Commission, (EC), High Level Group on Science Education, European Commission, & High Level Group on Science Education. (2007). *Science education now: A renewed pedagogy for the future of Europe* (Vol. 22845). Office for Official Publications of the European Communities.
- Falkner, K., Sentance, S., & Vivian, R. (n.d.). An International Study Piloting the MEasuring Teacher Enacted Computing Curriculum (METRECC) Instrument. *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education, 2019*, 111–142. <https://doi.org/10.1145/3344429.3372505>
- Falkner, K., Sentance, S., & Vivian, R. (2019). An International Benchmark Study of K-12 Computer Science Education in Schools. *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education.2019:257-258*.
- Falkner, K., Sentance, S., Vivian, R., Barksdale, S., Busuttil, L., Cole, E., Liebe, C., Maiorana, F., McGill, M. M., & Quille, K. (2019a). An international comparison of k-12 computer science education intended and enacted curricula. *Proceedings of the 19th Koli Calling International Conference on Computing Education Research*, 1–10.
- Falkner, K., Sentance, S., Vivian, R., Barksdale, S., Busuttil, L., Cole, E., Liebe, C., Maiorana, F., McGill, M. M., & Quille, K. (2019b). An international comparison of k-12 computer science education intended and enacted curricula. *Proceedings of the 19th Koli Calling International Conference on Computing Education Research*, 1–10.
- Falkner, K., Sentance, S., Vivian, R., Barksdale, S., Busuttil, L., Cole, E., Liebe, C., Maiorana, F., McGill, M. M., & Quille, K. (2019c). An international study piloting the measuring teacher enacted computing curriculum (metrecc) instrument.

Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education, 111–142.

- Falkner, K., Sentance, S., Vivian, R., Barksdale, S., Busuttil, L., Cole, E., Liebe, C., Maiorana, F., McGill, M. M., & Quille, K. (2019d). An international study piloting the measuring teacher enacted computing curriculum (metrece) instrument. In *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education* (pp. 111–142).
- Feaster, Y., Segars, L., Wahba, S. K., & Hallstrom, J. O. (2011). Teaching CS unplugged in the high school (with limited success). *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education*, 248–252.
- Fedorenko, E., Ivanova, A., Dhamala, R., & Bers, M. U. (2019). The language of programming: A cognitive perspective. *Trends in Cognitive Sciences*, 23(7), 525–528.
- Fincher, S. A., & Robins, A. V. (Eds.). (2019). *The Cambridge handbook of computing education research*. Cambridge University Press.
- Fincher, S., Robins, A., Baker, B., Box, I., Cutts, Q., Raadt, M., & Petre, M. (2006). Predictors of success in a first programming course. *Proceedings of the 8th Australasian Computing Education Conference (ACE 2006)*, 52, 189–196.
- Forlizzi, L., Lodi, M., Lonati, V., Mirolo, C., Monga, M., Montresor, A., Morpurgo, A., & Nardelli, E. (2018a). A core informatics curriculum for italian compulsory education. *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives*, 141–153.
- Forlizzi, L., Lodi, M., Lonati, V., Mirolo, C., Monga, M., Montresor, A., Morpurgo, A., & Nardelli, E. (2018b). A core informatics curriculum for italian compulsory education. *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives*, 141–153.

- Foster, J., & Connolly, R. (2017). Digital Media for STEM Learning: Developing scientific practice skills in the K-12 STEM classroom with resources from WGBH and PBS LearningMedia. *AGU Fall Meeting Abstracts*.
- Franke, B., & Osborne, B. (2015). Decoding CS principles: A curriculum from Code. org. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, 713–713.
- Franklin, D., Coenraad, M., Palmer, J., Eatinger, D., Zipp, A., Anaya, M., White, M., Pham, H., Gökdemir, O., & Weintrop, D. (2020). An Analysis of Use-Modify-Create Pedagogical Approach's Success in Balancing Structure and Student Agency. *Proceedings of the 2020 ACM Conference on International Computing Education Research*, 14–24.
- Freire, P. (2018a). *Pedagogy of the oppressed*. Bloomsbury publishing USA.
- Freire, P. (2018b). *Pedagogy of the oppressed*. Bloomsbury publishing USA.
- Frezza, S., Clear, T., & Clear, A. (2020). Unpacking Dispositions in the CC2020 Computing Curriculum Overview Report. *2020 IEEE Frontiers in Education Conference (FIE)*, 1–8.
- Frezza, S., Daniels, M., & Wilkin, A. (2019). Assessing students' IT professional values in a global project setting. *ACM Transactions on Computing Education (TOCE)*, 19(2), 1–34.
- Fuller, U., Johnson, C. G., Ahoniemi, T., Cukierman, D., Hernán-Losada, I., Jackova, J., Lahtinen, E., Lewis, T. L., Thompson, D. M., & Riedesel, C. (2007a). Developing a computer science-specific learning taxonomy. *ACM SIGCSE Bulletin*, 39(4), 152–170.
- Fuller, U., Johnson, C. G., Ahoniemi, T., Cukierman, D., Hernán-Losada, I., Jackova, J., Lahtinen, E., Lewis, T. L., Thompson, D. M., & Riedesel, C. (2007b). Developing a computer science-specific learning taxonomy. *ACM SIGCSE Bulletin*, 39(4), 152–170.

- Gal-Ezer, J., & Stephenson, C. (2014). A tale of two countries: Successes and challenges in K-12 computer science education in Israel and the United States. *ACM Trans Comput Educ*, 14(2). <https://doi.org/10.1145/2602483>
- Garcia, D. D., Harvey, B., & Segars, L. (2012a). CS principles pilot at University of California, Berkeley. *ACM Inroads*, 3(2), 58–60.
- Garcia, D. D., Harvey, B., & Segars, L. (2012b). CS principles pilot at University of California, Berkeley. *ACM Inroads*, 3, 58–60.
- Garcia, D., Harvey, B., & Barnes, T. (2015). The beauty and joy of computing. *ACM Inroads*, 6, 71–79.
- Giordano, D., & Maiorana, F. (n.d.). Teaching database: A pedagogical and curriculum perspective. *Proceedings of the International Conference on Information and Communication Technology for Education, (ICTE)*, 237–248.
- Giordano, D., & Maiorana, F. (2013a). An interdisciplinary project in sustainable development based on modern visual programming environments and web 2.0 technologies. *2013 3rd Interdisciplinary Engineering Design Education Conference*, 163–167.
- Giordano, D., & Maiorana, F. (2013b). Business Process Modeling and Implementation A 3-Year Teaching. *Experience. In Proceedings of the 5th International Conference on Computer Supported Education*, 429–436.
- Giordano, D., & Maiorana, F. (2013c). *Business Process Modeling and Implementation-A 3-Year Teaching Experience*. CSEDU.
- Giordano, D., & Maiorana, F. (2013d). Business Process Modeling and Implementation-A 3-Year Teaching Experience. *CSEDU*, 429–436.
- Giordano, D., & Maiorana, F. (2013e). Object oriented design through game development in XNA. *2013 3rd Interdisciplinary Engineering Design Education Conference*, 51–55.

- Giordano, D., & Maiorana, F. (2014a). Learning about the semantic web in an information systems oriented curriculum: A case study. *International Conference on Computer Supported Education*, 242–257.
- Giordano, D., & Maiorana, F. (2014b). Use of cutting edge educational tools for an initial programming course. *Global Engineering Education Conference (EDUCON)*, 556–563.
- Giordano, D., & Maiorana, F. (2014c). Use of cutting edge educational tools for an initial programming course. *2014 IEEE Global Engineering Education Conference (EDUCON)*, 556–563.
- Giordano, D., & Maiorana, F. (2014d). Use of cutting edge educational tools for an initial programming course. *2014 IEEE Global Engineering Education Conference (EDUCON)*, 556–563.
- Giordano, D., & Maiorana, F. (2015a). Teaching algorithms: Visual language vs flowchart vs textual language. *2015 IEEE Global Engineering Education Conference (EDUCON)*, 499–504.
- Giordano, D., & Maiorana, F. (2015b). Teaching algorithms: Visual language vs flowchart vs textual language. *2015 IEEE Global Engineering Education Conference (EDUCON)*, 499–504.
- Giordano, D., & Maiorana, F. (2013f). Teaching database: A pedagogical and curriculum perspective. *Proceedings of the International Conference on Information and Communication Technology for Education, (ICTE)*.
- Giordano, D., & Maiorana, F. (2013g). Teaching database: A pedagogical and curriculum perspective. *Proceedings of the International Conference on Information and Communication Technology for Education, (ICTE)*.
- Giordano, D., & Maiorana, F. (2014e). Teaching “design first” interleaved with object-oriented programming in a software engineering course. *2014 IEEE Global Engineering Education Conference (EDUCON)*, 1085–1088.

Giordano, D., Maiorana, F., Csizmadia, A., Marsden, S., Riedesel, C., & Mishra, S. (2015a).

New horizons in the assessment of computer science at school and beyond:

Leveraging on the ViVA platform. *ITiCSE-WGP 2015 - Proceedings of the 2015*

ITiCSE Conference on Working Group Reports. Association for Computing

Machinery, Inc, 117–147. <https://doi.org/10.1145/2858796.2858801>

Giordano, D., Maiorana, F., Csizmadia, A., Marsden, S., Riedesel, C., & Mishra, S. (2015b).

New horizons in the assessment of computer science at school and beyond:

Leveraging on the ViVA platform. *ITiCSE-WGP 2015 – Proceedings of the 2015*

ITiCSE Conference on Working Group Reports.

Giordano, D., Maiorana, F., Csizmadia, A. P., Marsden, S., Riedesel, C., Mishra, S., &

Vinikienė, L. (2015). New horizons in the assessment of computer science at school

and beyond: Leveraging on the viva platform. *Proceedings of the 2015 ITiCSE on*

Working Group Reports, 117–147.

Goldweber, M., Kaczmarczyk, L., & Blumenthal, R. (2019). Computing for the social good

in education. *ACM Inroads*, 10(4), 24–29.

Goncharow, A., Boekelheide, A., Mcquague, M., Burlinson, D., Saule, E., Subramanian, K.,

& Payton, J. (2019). Classifying pedagogical material to improve adoption of parallel

and distributed computing topics. *2019 IEEE International Parallel and Distributed*

Processing Symposium Workshops (IPDPSW), 312–319.

Gorman, J., Gsell, S., & Mayfield, C. (2014). Learning Relational Algebra by Snapping

Blocks. *DL.Acm.Org*, 2014, 73–78. <https://doi.org/10.1145/2538862.2538961>

Gotterbarn, D. W., Brinkman, B., Flick, C., Kirkpatrick, M. S., Miller, K., Vazansky, K., &

Wolf, M. J. (2018). ACM code of ethics and professional conduct.(2018). *URL*

Https://Www. Acm. Org/Code-of-Ethics.

Gove, M. (2012). *Michael Gove speech at the BETT Show 2012-Speeches Inside*

Government- GOV.UK. <https://www.gov.uk/government/speeches/michael->

- Gray, J., Abelson, H., Wolber, D., & Friend, M. (2012a). Teaching CS principles with app inventor. *Proceedings of the 50th Annual Southeast Regional Conference*, 405–406.
- Gray, J., Abelson, H., Wolber, D., & Friend, M. (2012b). Teaching CS principles with app inventor. *Proceedings of the 50th Annual Southeast Regional Conference*, 405–406.
- Griffin, P., & Care, E. (2014). *Assessment and Teaching of 21st Century Skills: Methods and Approach*. Springer.
- Guilford, J. P. (1967). Creativity: Yesterday, Today and Tomorrow. *The Journal of Creative Behavior*, 1(1), 3–14. <https://doi.org/10.1002/j.2162-6057.1967.tb00002.x>
- Guillén, M., Fontrodona, J., & Rodríguez-Sedano, A. (2007). The Great Forgotten Issue: Vindicating Ethics in the European Qualifications Framework (EQF). *Springer*, 74(4), 409–423. <https://doi.org/10.1007/s10551-007-9515-0>
- Gunion, K., Milford, T., & Stege, U. (2009). Curing recursion aversion. *ACM SIGCSE Bulletin*, 41(3), 124–128.
- Guzdial, M. (2015). Learner-centered design of computing education: Research on computing for everyone. *Synthesis Lectures on Human-Centered Informatics*, 8(6), 1–165.
- Guzdial, M. (2020a). Sizing the US student cohort for computer science. *Communications of the ACM*, 63(2), 10–11.
- Guzdial, M. (2020b). Sizing the U.S. student cohort for computer science. *Commun ACM*, 63(2), 10–11. <https://doi.org/10.1145/3374764>
- Guzdial, M., & Forte, A. (2005). Design process for a non-majors computing course. *ACM SIGCSE Bulletin*, 37(1), 361–365.
- Hadfield, S., & Schweitzer, D. (2009). Building an undergraduate computer science research experience. *2009 39th IEEE Frontiers in Education Conference*, 1-6,.
- Hamer, J., Cutts, Q., Jackova, J., Luxton-Reilly, A., McCartney, R., Purchase, H., & Sheard, J. (2008). Contributing student pedagogy. *ACM SIGCSE Bulletin*, 40, 194–212.

- Hamouda, S., Edwards, S. H., Elmongui, H. G., Ernst, J. V., & Shaffer, C. A. (2017). A basic recursion concept inventory. *Computer Science Education*, 27(2), 121–148.
- Han, M., Li, Z., He, J. S., & Tian, X. S. (2019). Understand the Emerging Demands of Computing Education for Non-CS Major Students. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 1266–1266.
- Harel, I. E., & Papert, S. E. (1991). *Constructionism*. Ablex Publishing.
- Harmse, A., & Wadee, A. A. (2019). Decolonizing ICT curricula in the era of the Fourth Industrial Revolution. *2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*, 1–10.
- Hazelkorn, E. (2015a). *Science education for responsible citizenship: Report to the European Commission of the Expert Group on Science Education*. Office of the European Union.
- Hazelkorn, E. (2015b). *Science education for responsible citizenship: Report to the European Commission of the Expert Group on Science Education*. Publications Office of the European Union.
- Hazelkorn, E. (2015c). *Science education for responsible citizenship: Report to the European Commission of the Expert Group on Science Education*. Publications Office of the European Union.
- Head, G. (2018). Ethics in educational research: Review boards, ethical issues and researcher development. *European Educational Research Journal*.
- Heatherton, T. F., & Wyland, C. L. (2003). *Assessing self-esteem*.
- Hello world. (2021). *The Big Book of Computing Pedagogy: Hello World's*. Raspberry Pi.
- Hodges, S., Sentance, S., Finney, J., & Ball, T. (2020). Physical computing: A key element of modern computer science education. *Computer*, 53(4), 20–30.
- Hoppe, H. U., & Werneburg, S. (2019). Computational thinking—More than a variant of scientific inquiry. *Computational Thinking Education*, 13–30.

- Hornbæk, K., & Hertzum, M. (2017). Technology acceptance and user experience: A review of the experiential component in HCI. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 24(5), 1–30.
- Howells, K. (2018a). *The future of education and skills: Education 2030: the future we want*.
- Howells, K. (2018b). The Future of Education and Skills: Education 2030: The Future We Want. In *Position Paper (05.04.2018).pdf*.
<http://create.canterbury.ac.uk/17331/1/E2030>
- Hsu, T., Chang, S., & YH-C&, E. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Elsevier*.
<https://www.sciencedirect.com/science/article/pii/S0360131518301799>.
- Hubley, A. M., & Zumbo, B. D. (2013). *Psychometric characteristics of assessment procedures: An overview*.
- Hubwieser, P., Armoni, M., Brinda, T., Dagiene, V., Diethelm, I., Giannakos, M. N., Knobelsdorf, M., Magenheimer, J., Mittermeir, R., & Schubert, S. (2011). Computer science/informatics in secondary education. *Proceedings of the 16th Annual Conference Reports on Innovation and Technology in Computer Science Education-Working Group Reports*, 19–38.
- Hubwieser, P., Giannakos, M. N., Berges, M., Brinda, T., Diethelm, I., Magenheimer, J., Pal, Y., Jackova, J., & Jasute, E. (2015). A global snapshot of computer science education in K-12 schools. In *Proceedings of the 2015 ITiCSE on working group reports* (pp. 65–83).
- Hubwieser, P., & Mühling, A. (n.d.). Playing PISA with bebras. *Association for Computing Machinery (ACM, 2014)*, 128–129. <https://doi.org/10.1145/2670757.2670759>
- Hug, S., Guenther, R., & Wenk, M. (2013). Cultivating a K12 computer science community: A case study. *SIGCSE 2013 - Proceedings of the 44th ACM Technical Symposium on Computer Science Education*, 275–280. <https://doi.org/10.1145/2445196.2445278>

- Hyseni Duraku, Z., & Hoxha, L. (2018). Self-esteem, study skills, self-concept, social support, psychological distress, and coping mechanism effects on test anxiety and academic performance. *Health Psychology Open*, 5(2), 2055102918799963.
- Ibe, N. A., Howsmon, R., Penney, L., Granor, N., Lyser, L. A., & Wang, K. (n.d.). Reflections of a diversity, equity, and inclusion working group based on data from a national CS education program. *SIGCSE 2018 - Proceedings of the 49th ACM Technical Symposium on Computer Science Education, 2018-January*, 711–716.
<https://doi.org/10.1145/3159450.3159594>
- Ibe, N. A., Howsmon, R., Penney, L., Granor, N., Lyser, L. A., & Wang, K. (2018). Reflections of a diversity, equity, and inclusion working group based on data from a national CS education program. *SIGCSE 2018 - Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 711–716.
<https://doi.org/10.1145/3159450.3159594>
- Iskander, M. F. (2018). Guest Editorial for the 25th Anniversary Special Issue of Computer Applications in Engineering Education Innovation in Engineering Education with Digital Technologies. *Comput Appl Eng Educ*, 26(5), 1066–1067.
<https://doi.org/10.1002/cae.22050>
- Izu, C., Mirolo, C., Settle, A., Mannila, L., & Stupurienė, G. (2017). *Exploring Bebras Tasks Content and Performance: A Multinational Study* (Vol. 16, Issue 1, pp. 39–59).
<https://doi.org/10.15388/infedu.2017.03>
- Izu, C., Schulte, C., Aggarwal, A., Cutts, Q., Duran, R., Gutica, M., Heinemann, B., Kraemer, E., Lonati, V., & Mirolo, C. (2019). Fostering program comprehension in novice programmers-learning activities and learning trajectories. In *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education* (pp. 27–52).

- Janitor, J., Jakob, F., & Kniewald, K. (2010). Visual learning tools for teaching/learning computer networks: Cisco networking academy and packet tracer. *2010 Sixth International Conference on Networking and Services*, 351–355.
- Jenkins, H. (2009). *Confronting the challenges of participatory culture: Media education for the 21st century*. The MIT Press.
- Jobin, A., Ienca, M., & Vayena, E. (2019). The global landscape of AI ethics guidelines. *Nature Machine Intelligence*, 1(9), 389–399.
- Johnson, C., & Bui, P. (2015). Blocks in, blocks out: A language for 3D models. In *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)* (pp. 77–82). IEEE.
- Johnson, C., McGill, M., & Bouchard, D. (2016). Game development for computer science education. *Proceedings of the 2016 ITiCSE Working Group Reports, ITiCSE 2016. Association for Computing Machinery, Inc*, 23–44.
<https://doi.org/10.1145/3024906.3024908>
- Johnson, C., Smith, R. S., Smythe, J. T., & Varon, R. K. (2009). *Challenge-Based Learning: An Approach for Our Time*. The New Media Consortium.
- Johnson, D. W., Johnson, R. T., & Stanne, M. B. (2000). *Cooperative learning methods: A meta-analysis*.
- Jormanainen, I. (2018). On computer science major students' motivation in a practically oriented robotics course. *Proceedings of the 18th Koli Calling International Conference on Computing Education Research*, 1–2.
- Kalelioglu, F., & Sentance, S. (2020). Teaching with physical computing in school: The case of the micro: bit. *Education and Information Technologies*, 25(4), 2577–2603.
- Kawas, S., Vonessen, L., & Ko, A. J. (2019a). Teaching accessibility: A design exploration of faculty professional development at scale. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 983–989.

- Kawas, S., Vonessen, L., & Ko, A. J. (2019b). Teaching accessibility: A design exploration of faculty professional development at scale. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 983–989.
- Kearney, C., & Gras-Velázquez, À. (2015). eTwinning Ten Years On: Impact on teachers' practice, skills, and professional development opportunities, as reported by eTwinners. *Brussels: Central Support Service of ETwinning European Schoolnet*.
- Kemp, P. E. J., & Berry, M. (2019). *The Roehampton Annual Computing Education Report: Pre-release snapshot from 2018*.
- Kitchin, R. (2014). Big Data, new epistemologies and paradigm shifts. *Big Data & Society*, 1(1), 2053951714528481.
- Ko, A. J., Beitlers, A., Wortzman, B., Davidson, M., Oleson, A., Kirdani-Ryan, M., & Druga, S. (2023). *Critically Conscious Computing: Methods for Secondary Education*. <https://criticallyconsciouscomputing.org/>
- Ko, A. J., Draper, S., Maguire, J., Pajunen, J., Tedre, M., Sinclair, J., & Szabo, C. (2023). A dialog about the special issues on theory. *ACM Transactions on Computing Education*, 23(1), 1–5.
- Kölling, M. (2018). Blue, bluej, greenfoot: Designing educational programming environments. In *Innovative Methods, User-Friendly Tools, Coding, and Design Approaches in People-Oriented Programming* (pp. 42–87). IGI Global.
- Kölling, M., Brown, N. C. C., Hamza, H., & McCall, D. (2019). Stride in bluej—Computing for all in an educational Ide. *SIGCSE 2019 - Proceedings of the 50th ACM Technical Symposium on Computer Science Education. Association for Computing Machinery, Inc*, 63–69. <https://doi.org/10.1145/3287324.3287462>
- Kong, S. C. (2019). Components and Methods of Evaluating Computational Thinking for Fostering Creative Problem-Solvers in Senior Primary School Education. In *Computational Thinking Education*. Springer Singapore. https://doi.org/10.1007/978-981-13-6528-7_8

- Kong, S. C., Chan, T.-W., & Griffin, P. (2014). E-learning in school education in the coming 10 years for developing 21st century skills: Critical research issues and policy implications. *J Educ Technol Soc*, 17(1), 70–78.
- Kong, S.-C., Abelson, H., & Lai, M. (2019). Introduction to Computational Thinking Education. In *Computational Thinking Education*. Springer Singapore.
https://doi.org/10.1007/978-981-13-6528-7_1
- Koschitz, D., & Rosenbaum, E. (2012). *EXPLORING ALGORITHMIC GEOMETRY WITH “BEETLE BLOCKS:” A GRAPHICAL PROGRAMMING LANGUAGE FOR GENERATING 3D FORMS*. 15 th International Conference on Geometry and Graphics. www.elica.net.
- Kotsopoulos, D., Floyd, L., Khan, S., Namukasa, I. K., Somanath, S., Weber, J., & Yiu, C. (2017). A pedagogical framework for computational thinking. *Digital Experiences in Mathematics Education*, 3, 154–171.
- Krishnamurthi, S., & Fisler, K. (2020). Data-centricity: A challenge and opportunity for computing education. *Communications of the ACM*, 63(8), 24–26.
- KSDE. Kansas state department of education., K. S. D. E. K. (2019). *Kansas Computer Science Standards. Grades P-12*. <https://www.ksde.org/Agency/Division-of-Learning-Services/Career-Standards-and-Assessment-Services/Content-Area-A-E/Computer-Science>
- Kumar, S., & Buyya, R. (2012). Green cloud computing and environmental sustainability. *Harnessing Green IT: Principles and Practices*, 315–339.
- Kundisch, D., Sievers, M., Zoyke, A., Herrmann, P., Whittaker, M., Beutner, M., & Magenheimer, J. (2012). *Designing a web-based application to support peer instruction for very large groups*.
- Ladner, R. E., & Stefik, A. (2017). AccessCSforall: Making computer science accessible to K-12 students in the United States. *ACM SIGACCESS Accessibility and Computing*, 118, 3–8.

- Larke, L. R. (2019). Agentic neglect: Teachers as gatekeepers of England's national computing curriculum. *British Journal of Educational Technology*, 50(3), 1137–1150.
- LeBlanc, C. A., Newcomb, J., & Rheingans, P. (2020). First Generation-Rural Computer Science Students. *Association for Computing Machinery (ACM)*, 1360–1360. <https://doi.org/10.1145/3328778.3372662>
- Lee, E., Shan, V., Beth, B., & Lin, C. (2014). A structured approach to teaching recursion using cargo-bot. *Proceedings of the Tenth Annual Conference on International Computing Education Research*, 59–66.
- Lemay, D., & Basnet, R. (2017). Algorithmic thinking, cooperativity, creativity, critical thinking, and problem solving: Exploring the relationship between computational thinking skills and academic performance. *Journal of Computers in Education*, 4(4), 355–369. <https://doi.org/10.1007/s40692-017-0090-9>
- Levitin A., & Levitin M. (2011). *Algorithmic Puzzles*. OUP USA.
- Licht, A. H., Tasiopoulou, E., & Wastiau, P. (2017a). *Open book of educational innovation*. European Schoolnet.
- Licht, A. H., Tasiopoulou, E., & Wastiau, P. (2017b). *Open book of educational innovation*. Brussels: European Schoolnet.
- Linebarger, D. L. (2000). *Summative evaluation of Between the Lions: A final report to WGBH Educational Foundation* (Vol. 3, Issue 2010).
- Liu, C.-C., Chen, H. S. L., Shih, J.-L., Huang, G.-T., & Liu, B.-J. (2011). An enhanced concept map approach to improving children's storytelling ability. *Comput Educ*, 56(3), 873–884.
- Looi, C. K., How, M. L., Longkai, W., Seow, P., & Liu, L. (2018). Analysis of linkages between an unplugged activity and the development of computational thinking. *Comput Sci Educ*, 28(3), 255–279. <https://doi.org/10.1080/08993408.2018.1533297>.
- Looi, C.-K., Lim, W.-Y., & Chen, W. (2008). Communities of practice for continuing professional development in the twenty-first century. In *International Handbook of*

Springer.

- Looi, C.-K., Sun, D., Seow, P., & Chia, G. (2014). *Enacting a technology-based science curriculum across a grade level: The journey of teachers' appropriation the Creative Commons Attribution-Non-Commercial-No-Derivatives (CC-BY-NC-ND) 4.0 license*. <https://doi.org/10.1016/j.compedu.2013.10.006>
- Luxton-Reilly, A., Becker, B. A., Cao, Y., McDermott, R., Mirolo, C., Mühling, A., & Whalley, J. (2018). Developing assessments to determine mastery of programming fundamentals. *Proceedings of the 2017 ITiCSE Conference on Working Group Reports*, 47–69.
- Magana, A. J., & Jong, T. (2018). Modeling and simulation practices in engineering education. *Comput Appl Eng Educ*, 26(4), 731–738. <https://doi.org/10.1002/cae.21980>
- Maiorana, F. (2014). Extending the Database Curriculum: From Design Principles to Web and Mobile Programming. *International Conference on Computer Supported Education*, 258–271.
- Maiorana, F. (2015). Extending the database curriculum: From design principles to web and mobile programming. *International Conference on Computer Supported Education*, 258–271.
- Maiorana, F. (2018). *Computational Thinking and Humanities* (pp. 87–96).
- Maiorana, F. (2019a). Interdisciplinary Computing for STE (A) M: a low Floor high ceiling curriculum. *Innov Technol Res Educ*, 37.
- Maiorana, F. (2019b). Interdisciplinary Computing for STE (A) M: A low Floor high ceiling curriculum. *Innovations, Technologies and Research in Education*, 37.
- Maiorana, F. (2019c). Interdisciplinary Computing for STE (A) M: A low Floor high ceiling curriculum. *Innovations, Technologies and Research in Education*, 37.

- Maiorana, F. (2021a). A flipped design of learning resources for a course on algorithms and data structures. *International Conference on Interactive Collaborative and Blended Learning*, 268–279.
- Maiorana, F. (2021b). A flipped design of learning resources for a course on algorithms and data structures. *International Conference on Interactive Collaborative and Blended Learning*, 268–279.
- Maiorana, F. (2021c). From High School to Higher Education: Learning Trajectory for an Inclusive and Accessible Curriculum for Teachers and Their Students. *International Journal of Smart Education and Urban Society (IJSEUS)*, 12(4), 36–51.
- Maiorana, F. (2021d). From High School to Higher Education: Learning Trajectory for an Inclusive and Accessible Curriculum for Teachers and Their Students. *International Journal of Smart Education and Urban Society (IJSEUS)*, 12(4), 36–51.
- Maiorana, F. (2021e). From High School to Higher Education: Learning Trajectory for an Inclusive and Accessible Curriculum for Teachers and Their Students. *International Journal of Smart Education and Urban Society (IJSEUS)*, 12(4), 36–51.
- Maiorana, F. (2020). A flipped design of learning resources for a course on algorithms and data structures. *Proceedings of the 9th International Conference on Interactive Collaborative and Blended Learning*.
- Maiorana, F., Altieri, S., Colli, A., Labbri, M., Nicolini, M., Nazzaro, L., Porta, M., Severi, A., & Guida, M. (2020). SCIENTIX TEACHER AMBASSADORS: A PASSIONATE AND CREATIVE PROFESSIONAL COMMUNITY LINKING RESEARCH AND PRACTICE. *Proceedings of ICERI2020 Conference*, 9, 10th.
- Maiorana, F., Berry, M., Nelson, M., Lucarelli, C., Phillipps, M., Mishra, S., & Benassi, A. (2017a). International perspectives on CS teacher formation and professional development. *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education*, 236–237.

- Maiorana, F., Berry, M., Nelson, M., Lucarelli, C., Phillipps, M., Mishra, S., & Benassi, A. (2017b). International perspectives on CS teacher formation and professional development. *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education*, 236–237.
- Maiorana, F., Berry, M., Nelson, M., Lucarelli, C., Phillipps, M., Mishra, S., & Benassi, A. (2017c). International perspectives on CS teacher formation and professional development. *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education*, 236–237.
- Maiorana, F., & Cristaldi, G. (n.d.). From Data to Coding and responsible digital citizenship: The design of a learning journey. In *Surveys on students: The INVALSI national and international tests. VI Seminar 'INVALSI data: A tool for teaching and scientific research'*.
- Maiorana, F., Csizmadia, A. P., & Richards, G. M. (2020a). Managing Data and Projects: Lessons Learnt from Comparing Computing Curricula. *2020 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, 165–172.
- Maiorana, F., Csizmadia, A. P., & Richards, G. M. (2020b). P12 Computing in Italy, England and Alabama, USA. *Proceedings of the 21st Annual Conference on Information Technology Education*, 242–247.
- Maiorana, F., Csizmadia, A. R., G., & Riedesel, C. (2020). Recursion and iteration: A combined approach for algorithm comprehension. *Proceedings of the 9th International Conference on Interactive Collaborative and Blended Learning*.
- Maiorana, F., Csizmadia, A., Richards, G., & Riedesel, C. (2021). Recursion Versus Iteration: A Comparative Approach for Algorithm Comprehension. *International Conference on Interactive Collaborative and Blended Learning*, 247–259.
- Maiorana, F., Csizmadia, A., & Richards, R. (2023). Computer Systems and HCI perspective in Teaching Computing in England, Italy and USA. *Informatics in Education*.

- Maiorana, F., & Giordano, D. (2013a). A constructivist approach to teaching index selection strategies and database design. *Proceedings of the International Conference on Information and Communication Technology for Education, (ICTE)*.
- Maiorana, F., & Giordano, D. (2013b). A constructivist approach to teaching index selection strategies and database design. *Proceedings of the International Conference on Information and Communication Technology for Education, (ICTE)*.
- Maiorana, F., Giordano, D., & Morelli, R. (2015a). Quizly: A live coding assessment platform for App Inventor. In *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)* (pp. 25–30).
- Maiorana, F., Giordano, D., & Morelli, R. (2015b). Quizly: A live coding assessment platform for App Inventor. *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)*, 25–30.
- Maiorana, F., Gras-Velazquez, A., Paladino, E., Cristaldi, G., Niewint-Gori, J., Mishra, S., Trifas, M., Sharmin, S., & Gonzalez, C. (2023). *Coding, Computational, Algorithmic, Design, Creative, and Critical Thinking in K-16 education*. In press.
- Maiorana, F., Nazzaro, L., Severi, A., Ciolli, A., Porta, M., Cristaldi, G., & Labbri, M. (2022a). Reflections on inclusive leadership education: From professional communities of practices to students. *IUL Research*, 3(5), 324–337.
- Maiorana, F., Nazzaro, L., Severi, A., Ciolli, A., Porta, M., Cristaldi, G., & Labbri, M. (2022b). Reflections on inclusive leadership education: From professional communities of practices to students. *IUL Research*, 3(5), 324–337.
- Maiorana, F., Richards, G., Lucarelli, C., Berry, M., & Ericson, B. (2019a). Interdisciplinary Computer Science Pre-service TeacherPreparation: Panel. *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*, 332–333.
- Maiorana, F., Richards, G., Lucarelli, C., Berry, M., & Ericson, B. (2019b). Interdisciplinary Computer Science Pre-service TeacherPreparation: Panel. *Proceedings of the 2019*

ACM Conference on Innovation and Technology in Computer Science Education,
332–333.

Maiorana, F., Richards, G., Lucarelli, C., Miles, B., & Ericson, B. (2019). Interdisciplinary Computer Science pre-service Teacher Preparation. *24th Annual Conference on Innovation and Technology in Computer Science Education*.

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014a). Computational thinking in K-9 education. *Proceedings of the Working Group Reports of the 2014 on Innovation & Technology in Computer Science Education Conference*, 1–29.

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014b). Computational thinking in K-9 education. *Proceedings of the Working Group Reports of the 2014 on Innovation & Technology in Computer Science Education Conference*, 1–29.

Martin, F., Lee, I., Lytle, N., Sentance, S., & Lao, N. (2020). Extending and evaluating the use-modify-create progression for engaging youth in computational thinking. *Proceedings of the 51st Acm Technical Symposium on Computer Science Education*, 807–808.

Martin, J., Edwards, S. H., & Shaffer, C. A. (2015). The effects of procrastination interventions on programming project success. *Proceedings of the Eleventh Annual International Conference on International Computing Education Research*, 3–11.

Maslow, A., & Lewis, K. J. (1987). Maslow's hierarchy of needs. *Salenger Incorporated*, 14(17), 987–990.

McCauley, R., Grissom, S., Fitzgerald, S., & Murphy, L. (2015). Teaching and learning recursive programming: A review of the research literature. *Comput Sci Educ*, 25(1), 37–66. <https://doi.org/10.1080/08993408.2015.1033205>

McConnell, J. J. (1996). Active learning and its use in computer science. *Proceedings of the 1st Conference on Integrating Technology into Computer Science Education*, 52–54.

- Mcgettrick, A., Boyle, R., Ibbett, R., Lloyd, J., Lovegrove, G., & Mander, K. (2005). Grand Challenges Grand Challenges in Computing: Education-A Summary. *Comput J*, 48(1). <https://doi.org/10.1093/comjnl/bxh064>
- McGill, M. M., Johnson, C., & Atlas, J. (2018). If memory serves: Towards designing and evaluating a game for teaching pointers to undergraduate students. *ITiCSE-WGR 2017 - Proceedings of the 2017 ITiCSE Conference on Working Group Reports*, 25–46. <https://doi.org/10.1145/3174781.3174783>
- McGill, M. M., Snow, E., & Camping, A. (2022). A Theory of Impacts Model for Assessing Computer Science Interventions through an Equity Lens: Identifying Systemic Impacts Using the CAPE Framework. *Education Sciences*, 12(9), 578.
- McGrew, S. (2020). Learning to evaluate: An intervention in civic online reasoning. *Computers & Education*, 145, 103711.
- McGrew, S., & Byrne, V. L. (2020). Who Is behind this? Preparing high school students to evaluate online content. *Journal of Research on Technology in Education*, 53(4), 457–475.
- McLeod, S. (2007). Maslow's hierarchy of needs. *Simply Psychology*, 1(1–18).
- McQuaigue, M., Beckman, A., Burlinson, D., Sloop, L., Goncharow, A., Saule, E., Subramanian, K., & Payton, J. (2020). An Engaging CS1 Curriculum Using BRIDGES. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 1317–1317.
- Miller, C. S., & Settle, A. (2011). When practice doesn't make perfect: Effects of task goals on learning computing concepts. *ACM Trans Comput Educ*, 11(4). <https://doi.org/10.1145/2048931.2048933>
- Miller, L. D., Soh, L.-K., Chiriacescu, V., Ingraham, E., Shell, D. F., Ramsay, S., & Hazley, M. P. (2013). Improving learning of computational thinking using creative thinking exercises in CS-1 computer science courses. *2013 Ieee Frontiers in Education Conference (Fie)*, 1426–1432.

- Milne, L. R., & Ladner, R. E. (2019a). Position: Accessible Block-Based Programming: Why and How. In *2019 IEEE Blocks and Beyond Workshop (B&B)*.
- Milne, L. R., & Ladner, R. E. (2019b). Position: Accessible Block-Based Programming: Why and How. In *2019 IEEE Blocks and Beyond Workshop (B&B)* (pp. 19–22).
- Mitchell, J. (2020). Editorial: Transactions on Education in the New Decade. *IEEE Transactions on Education*, *63*(1), 1–1. <https://doi.org/10.1109/TE.2021.3071049>.
- Mitchell, J. (2021). Editorial: Transactions on Education. *IEEE Transactions on Education*, *64*(2), 87–87,. <https://doi.org/10.1109/TE.2021.3071049>.
- Mitsea, E., Drigas, A., & Skianis, C. (2023). VR Gaming for Meta-Skills Training in Special Education: The Role of Metacognition, Motivations, and Emotional Intelligence. *Education Sciences*, *13*(7), 639.
- Molloy, J. C. (2011). The open knowledge foundation: Open data means better science. *PLoS Biol*, *9*(12).
- Monge, A. E., Fadjo, C. L., Quinn, B. A., & Barker, L. J. (2015). EngageCSEdu: Engaging and retaining CS1 and CS2 students. *ACM Inroads*, *6*(1), 6–11. <https://doi.org/10.1145/2714569>
- Monig, J., Ohshima, Y., & Maloney, J. (2015). Blocks at your fingertips: Blurring the line between blocks and text in GP. In *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)* (pp. 51–53).
- Morelli, R. A., Wolber, D., Rosato, J., Uche, C., & Lake, P. (2014). Mobile computer science principles: A professional development sampler for teachers. *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, 750–750.
- Morelli, R., Tucker, A., & Danner, N. (2009). Revitalizing computing education through free and open source software for humanity. *Commun ACM*, *52*(8), 67–75. <https://doi.org/10.1145/1536616.1536635>
- Moreno-León, J., Robles, G., Román-González, M., J, M.-L., & RománRom, M. (2017). Towards Data-Driven Learning Paths to Develop Computational Thinking with

- Scratch. In *IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING* 2017.
<https://doi.org/10.1109/TETC.2017.2734818>
- Müller, V. C. (n.d.). Philosophy of AI: A structured overview. Forthcoming in Smüha. In Nathalie (Ed.), (2024): *Cambridge Handbook on the Law, Ethics and Policy of Artificial Intelligence*. Cambridge University Press.
- Myller, N., Bednarik, R., & Moreno, A. (2007). Integrating dynamic program visualization into BlueJ: The Jeliot 3 extension. *Seventh IEEE International Conference on Advanced Learning Technologies (ICALT 2007)*, 505–506.
- Nardelli, E., Forlizzi, L., Lodi, M., Lonati, V., Mirolo, C., Monga, M., Montessor, A., & Morpugo, A. (2017). *Proposal for a national Informatics curriculum in the Italian school*. National Interuniversity consortium for Informatics. CINI.
- Nardelli, E., Forlizzi, L., Lodi, M., Lonati, V., Mirolo, C., Monga, M., Montresor, A., & Morpurgo, A. (2017a). *Proposal for a national Informatics curriculum in the Italian school* [Technical Report. CINI.]. <https://www.>
- Nardelli, E., Forlizzi, L., Lodi, M., Lonati, V., Mirolo, C., Monga, M., Montresor, A., & Morpurgo, A. (2017b). *Proposal for a national Informatics curriculum in the Italian school*. Technical Report. CINI. [https://www. consortio-cini. it/images/PROPOSAL](https://www.consortio-cini.it/images/PROPOSAL)
....
- Nature. (2010). *Education ambivalence*. 525–526.
- Naval, C., Villacís, J. L., & Ibarrola-García, S. (2022). The transversality of civic learning as the basis for development in the university. *Education Sciences*, 12(4), 240.
- Nieh, J., & Vaill, C. (2005). Experiences teaching operating systems using virtual platforms and linux. *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education*, 520–524.
- NIST. (2014). Assessing Security and Privacy Controls in Federal Information Systems and Organizations. *NIST Special Publication (SP)*.
<http://dx.doi.org/10.6028/NIST.SP.800-53Ar4> . 2014

- Nistor, A., Clemente-Gallardo, J., Angelopoulos, T., Chodzinska, K., Clemente Gallardo, M., Gozdzik, A., Gras-Velazquez, A., Grizelj, A., Kolenberg, K., & Mitropoulou, D. (2019). Bringing Research into the Classroom—The Citizen Science approach in schools. *Scientix Observatory*.
- Nistor, A., Gras-Velazquez, A., Billon, N., & Mihai, G. (2018). *Science, Technology, Engineering and Mathematics Education Practices in Europe* [Scientix Observatory report,]. European Schoolnet.
- Novak, J. D., & Cañas, A. J. (2006). The theory underlying concept maps and how to construct them. *Florida Inst Hum Mach Cogn*, 1(1), 1–31.
- Nussloch, R. H., Preston, J., & Reichgelt, H. (2014). *Improving student success through personalization and customization*.
- Nylén, A., Daniels, M., Isomöttönen, V., & McDermott, R. (2017). Open-ended projects opened up—Aspects of openness. *2017 IEEE Frontiers in Education Conference (FIE)*, 1–7.
- Oates, T., Coe, R., Peyton Jones, S., Scratcherd, T., & Woodhead, S. (2016). *Quantum: Tests worth teaching to*.
- Oates, T., Coe, R., Peyton-Jones, S., Scratcherd, T., & Woodhead, S. (2016a). *Quantum: Tests worth teaching. White Paper*.
- Oates, T., Coe, R., Peyton-Jones, S., Scratcherd, T., & Woodhead, S. (2016b). *Quantum: Tests worth teaching. White Paper, March, Computing at School*.
- O.E.C.D. (2021). *PISA 2021 Creative Thinking Framework* (Vol. 53, pp. 1689–1699).
- OECD learning. (2030). *OECD learning compass 2030. COMPASS*.
www.oecd.org/education/2030-project.
- OECD Publishing. (2021). *21st-century readers: Developing literacy skills in a digital world*. Organisation for Economic Co-operation and Development OECD.

- Osborne, H., & Yurcik, W. (2002). The educational range of visual simulations of the little man computer architecture paradigm. *32nd Annual Frontiers in Education, 3*, S4G-S4G.
- Owens, T. L. (2017). Higher education in the sustainable development goals framework. *European Journal of Education, 52*(4), 414–420.
- Ozmen Garibay, O., Winslow, B., Andolina, S., Antona, M., Bodenschatz, A., Coursaris, C., & Xu, W. (2023). Six human-centered artificial intelligence grand challenges. *International Journal of Human–Computer Interaction, 39*(3), 391–437.
- Papadakis, S. (2016). Creativity and innovation in European education. Ten years eTwinning. Past, present and the future. *Int J Technol Enhanc Learn, 8*(3–4), 279–296.
- Park, J. H., Kim, H.-Y., & Lim, S.-B. (2019a). Development of an electronic book accessibility standard for physically challenged individuals and deduction of a production guideline. *Comput Stand Interfaces, 64*, 78–84.
- Park, J. H., Kim, H.-Y., & Lim, S.-B. (2019b). Development of an electronic book accessibility standard for physically challenged individuals and deduction of a production guideline. *Comput Stand Interfaces, 64*, 78–84.
- Patton, E., Tissenbaum, M., & Harunani, F. (2019). MIT App Inventor: Objectives, Design, and Development. In E. Patton, M. Tissenbaum, & F. Harunani (Eds.), *Computational Thinking Education* (pp. 31–49). Springer.
- Patton, E. W., Tissenbaum, M., & Harunani, F. (2019). MIT app inventor: Objectives, design, and development. In *Computational thinking education* (pp. 31–49). Springer, Singapore.
- Payton, F. C., White, A., & Mullins, T. (2017). STEM majors, art thinkers (STEM+arts)-Issues in duality, rigor, and inclusion. *Journal of STEM Education, 18*(3), 39–47.
- Pearce, J. M. (2012). Building Research Equipment with Free, Open-Source Hardware MATERIALS SCIENCE. *Science.Sciencemag.Org*.
<https://doi.org/10.1126/science.1224989>

- Pecanin, E., Spalevic, P., Mekic, E., Jovic, S., & Milovanovic, I. (2019). E-learning engineers based on constructive and multidisciplinary approach. *Comput Appl Eng Educ*, 27(6), 1544–1554. <https://doi.org/10.1002/cae.22168>
- Peña-López, I. (2017). *PISA 2015 results (Volume III). Students' well-being: Vol. III. Well-Being*.
- Peña-López, I. (2019). *PISA 2018 Results. What School Life Means for Students' Lives*.
- Pérez, A., Potocki, A., Stadtler, M., Macedo-Rouet, M., Paul, J., Salmerón, L., & Rouet, J.-F. (2018). Fostering teenagers' assessment of information reliability: Effects of a classroom intervention focused on critical source dimensions. *Learning and Instruction*, 58, 53–64.
- Piwek, P., & Savage, S. (2020). Challenges with Learning to Program and Problem Solve. *An Analysis of Student Online Discussions Conference or Workshop Item Challenges with Learning to Program and Problem Solve: An Analysis of Student Online Discussions*. <https://doi.org/10.1145/3328778.3366838>
- PMIEF. (n.d.). *Reflecting on Project Management for Social Good*. Project Management Institute Educational Foundation. [https:// pmief.org/pm-for-social-good](https://pmief.org/pm-for-social-good)
- Pollock, I., Alshaigy, B., Bradley, A., Krogstie, B. R., Kumar, V., Ott, L., Peters, A.-K., Riedesel, C., & Wallace, C. (2019). 1.5 Degrees of Separation: Computer Science Education in the Age of the Anthropocene. *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education*, 1–25.
- Poongodi, T., Ramya, S. R., Suresh, P., & Balusamy, B. (2020). Application of IoT in green computing. In *Advances in Greener Energy Technologies* (pp. 295–323). Springer.
- Porter, A. C., & Smithson, J. L. (2001a). *Defining, Developing, and Using Curriculum Indicators* [CPRE Research Report Series.].
- Porter, A. C., & Smithson, J. L. (2001b). *Defining, Developing, and Using Curriculum Indicators*. *CPRE Research Report Series*.

- Porter, L., Bouvier, D., Cutts, Q., Grissom, S., Lee, C., McCartney, R., Zingaro, D., & Simon, B. (2016a). A multi-institutional study of peer instruction in introductory computing. *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, 358–363.
- Porter, L., Bouvier, D., Cutts, Q., Grissom, S., Lee, C., McCartney, R., Zingaro, D., & Simon, B. (2016b). A multi-institutional study of peer instruction in introductory computing. *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, 358–363.
- Porter, L., Zingaro, D., Lee, C., Taylor, C., Webb, K. C., & Clancy, M. (2018). Developing course-level learning goals for basic data structures in CS2. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 858–863.
- Prasad, S., Gupta, A., Rosenberg, A., Sussman, A., & Weems, C. (2015). *Topics in parallel and distributed computing*. Springer.
- Prasad, S. K., Gupta, A., Rosenberg, A., Sussman, A., & Weems, C. (2018). *Topics in Parallel and Distributed Computing: Enhancing the Undergraduate Curriculum: Performance, Concurrency, and Programming on Modern Platforms*. Springer.
- Prince, M. (2004). Does active learning work? A review of the research. *Journal of Engineering Education*, 93(3), 223–231.
- Qian, M., & Clark, K. R. (2016). Game-based Learning and 21st century skills: A review of recent research. *Comput Human Behav*, 63, 50–58.
- Qin, Z., Johnson, D. W., & Johnson, R. T. (1995). Cooperative versus competitive efforts and problem solving. *Review of Educational Research*, 65(2), 129–143.
- Quille, K., & Bergin, S. (2019). CS1: How will they do? How can we help? A decade of research and practice. *Computer Science Education*, 29(2–3), 254–282.
- Quille, K., McGill, M., Vivian, R., Falkner, K., Sentance, S., Barksdale, S., Busuttill, L., Cole, E., Liebe, C., & Maiorana, F. (2020, June). An International Pilot Study of K-12 Teachers' Computer Science Self-Esteem. *Proceedings of 25th ACM International*

Conference on Innovation and Technology in Computer Science Education. ACM. In Press.

- Repenning, A., Basawapatna, A., & Escherle, N. (n.d.). Computational thinking tools. *2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC, 2016)*, 218–222.
- Ribble, M., & Bailey, G. (2007). *Digital Citizenship in Schools, Mike Ribble and Gerald Bailey. 1.800.336.5191 or 1.541* (Vol. 302).
www.infoworld.com/article/05/11/22/hnonlineshoppers_1.html.
- Rich, K. M., Carla Strickland, T. A. B., Moran, C., & Franklin, D. (2017). K-8 Learning Trajectories Derived from Research Literature: Sequence, Repetition, Conditionals. *Proceedings of the 2017 ACM Conference on International Computing Education Research (ICER '17)*. ACM, 182–190.
- Richards, G., & Turner, T. E. (2019). Infusing Cybersecurity Concepts into PK-12 Education: The Complexity of Integrating Multiple Standards. Retrieved November, 2, 2021.
- Rinderknecht, C. (2014). *A Survey on Teaching and Learning Recursive Programming* (Vol. 13). <https://www.ceeol.com/search/article-detail?id=123696>.
- Rode, J. A., Weibert, A., & Marshall, A. (2015). From computational thinking to computational making. *UbiComp 2015 - Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing. Association for Computing Machinery, Inc*, 239–250. <https://doi.org/10.1145/2750858.2804261>
- Rogers-Shaw, C., Carr-Chellman, D. J., & Choi, J. (2018). Universal design for learning: Guidelines for accessible online instruction. *Adult Learn*, 29(1), 20–31.
- Román-González, M., Moreno-León, J., & Robles, G. (2019). Combining Assessment Tools for a Comprehensive Evaluation of Computational Thinking Interventions. *Computational Thinking Education. Springer Singapore*, 79–98.
https://doi.org/10.1007/978-981-13-6528-7_6

- Román-González, M., Pérez-González, J. C., Moreno-León, J., & Robles, G. (2018). Can computational talent be detected? Predictive validity of the Computational Thinking Test. *International Journal of Child-Computer Interaction*, 18, 47–58.
- Romero, M., Lepage, A., & Lille, B. (2017). Computational thinking development through creative programming in higher education. *Int J Educ Technol High Educ*, 14(1).
<https://doi.org/10.1186/s41239-017-0080-z>
- Rosato, J., Lucarelli, C., Beckworth, C., & Morelli, R. (2017a). A comparison of online and hybrid professional development for cs principles teachers. *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education*, 140–145.
- Rosato, J., Lucarelli, C., Beckworth, C., & Morelli, R. (2017b). A comparison of online and hybrid professional development for cs principles teachers. *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education*, 140–145.
- Rosenberg, M. (2015). *Society and the adolescent self-image*. Princeton university press.
- Ross, J., & Freeman, M. (2022). *Client-Side Web Development* (2022nd ed.).
<https://info340.github.io/index.html>
- Rowland, A., Dawkins, T., Risler, J., & Ayers, T. (2018). The Value of Industry Certifications in ITE: Panel Discussion. *Proceedings of the 19th Annual SIG Conference on Information Technology Education*, 84–85.
- Royal Society. (2012). *Shutdown or restart? The way forward for computing in UK schools*. Royal Society. <https://royalsociety.org/topics-policy/projects/computing-inschools/report/>.
- Royal Society. (2017). After the reboot: Computing education in UK schools. *Policy Report*.
- Russell, S. H., Hancock, M. P., & McCullough, J. (2007). Benefits of undergraduate research experience. *Benefits of Undergraduate Research Experiences*.

- Saleh, I. M., & Khine, M. S. (2023). *New Science of Learning: Exploration in Mind, Brain, and Education* (Vol. 14).
- Santo, R., Vogel, S., DeLyser, L. A., & Ahn, J. (2018). Asking "CS4What?" as a basis for CS4All: Workshop tools to support sustainable K-12 CS implementations. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 678–686.
- Savage, M., & Csizmadia, A. (2018). Computational thinking and creativity in the secondary curriculum. In *Debates in Computing and ICT Education*. Routledge (pp. 137–152). <https://doi.org/10.4324/9781315709505-10>
- Scherer, R., Siddiq, F., & Sánchez Viveros, B. (2019). The cognitive benefits of learning computer programming: A meta-analysis of transfer effects. *Journal of Educational Psychology*, 111(5), 764.
- Schrire, S., & Levy, D. (2012). Troubleshooting MOOCs: The case of a massive open online course at a college of education. *EdMedia+ Innovate Learning*, 761–766.
- Schulz, W., Ainley, J., Fraillon, J., Losito, B., Agrusti, G., & Friedman, T. (2018). *Becoming citizens in a changing world: IEA International Civic and Citizenship Education Study 2016 international report*. Springer Nature.
- Seehorn, D., Carey, S., Fuschetto, B., Lee, I., Moix, D., O'Grady-Cunniff, D., Owens, B. B., Stephenson, C., & Verno, A. (2011). *CSTA K–12 Computer Science Standards: Revised 2011*. ACM.
- Sentance, S., & Csizmadia, A. (2015). Teachers' perspectives on successful strategies for teaching Computing in school. In *IFIP TCS*.
- Sentance, S., & Csizmadia, A. (2017a). Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Educ Inf Technol*, 22(2), 469–495. <https://doi.org/10.1007/s10639-016-9482-0>

- Sentance, S., & Csizmadia, A. (2017b). Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Education and Information Technologies*, 22(2), 469–495.
- Sentance, S., & Csizmadia, A. (2017c). Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Education and Information Technologies*, 22(2), 469–495.
- Sentance, S., Singh, L., & Freitas, P. (2020). Challenges Facing Computing Teachers in Guyana. *Association for Computing Machinery (ACM)*, 1323–1323.
<https://doi.org/10.1145/3328778.3372613>
- Sentance, S., & Thota, N. (2013a). A comparison of current trends within Computer Science teaching in school in Germany and the UK. *Informatics in Schools: Local Proceedings of the 6th International Conference ISSEP 2013; Selected Papers; Oldenburg, Germany, February 26–March 2, 2013*, 6, 63.
- Sentance, S., & Thota, N. (2013b). A comparison of current trends within Computer Science teaching in school in Germany and the UK. *Informatics in Schools: Local Proceedings of the 6th International Conference ISSEP 2013*, 6, 63.
- Sentance, S., & Waite, J. (2017). PRIMM: Exploring pedagogical approaches for teaching text-based programming in school. *Proceedings of the 12th Workshop on Primary and Secondary Computing Education*, 113–114.
- Sentance, S., Waite, J., & Kallia, M. (2019a). Teachers' experiences of using primm to teach programming in school. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 476–482.
- Sentance, S., Waite, J., & Kallia, M. (2019b). Teaching computer programming with PRIMM: A sociocultural perspective. *Computer Science Education*, 29(2–3), 136–176.
- Seow, P., Looi, C.-K., How, M.-L., Wadhwa, B., & Wu, L.-K. (2019). Educational Policy and Implementation of Computational Thinking and Programming: Case Study of

- Singapore. *Computational Thinking Education*. Springer Singapore, 345–361.
https://doi.org/10.1007/978-981-13-6528-7_19
- Sharp, A., & McDermott, P. (2009). *Workflow modeling: Tools for process improvement and applications development*. Artech House.
- Sharples, M., Roock, R., & Ferguson, R. (2016). Innovating Pedagogy 2016 Exploring New Forms of Teaching. *Learning and Assessment, to Guide Educators and Policy Makers*. www.open.ac.uk/innovating.
- Shavelson, R. J., & Bolus, R. (1982). Self concept: The interplay of theory and methods. *Journal of Educational Psychology*, 74(1), 3.
- Shih, J., Shih, B., Shih, C., Su, H., & CC-C&, E. (2010). The influence of collaboration styles to children's cognitive performance in digital problem-solving game "William Adventure": A comparative case study. *Elsevier*.
<https://www.sciencedirect.com/science/article/pii/S0360131510001168>.
- Shih, J.-L., Nuutinen, J., Hwang, G.-J., & Chen, N.-S. (2010). Building virtual collaborative research community using knowledge management approach. *Knowl Manag E-Learning An Int J*, 2(3), 293–311.
- Shinohara, K., Kawas, S., Ko, A. J., & Ladner, R. E. (2018a). Who teaches accessibility? A survey of US computing faculty. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 197–202.
- Shinohara, K., Kawas, S., Ko, A. J., & Ladner, R. E. (2018b). Who teaches accessibility? A survey of US computing faculty. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 197–202.
- Simon, B., & Cutts, Q. (2012). Peer instruction: A teaching method to foster deep understanding. *Communications of the ACM*, 55(2), 27–29.
- Simon, S. J., & M, M. (2016). Negotiating the maze of academic integrity in computing education. *Proceedings of the 2016 ITiCSE Working Group Reports, ITiCSE 2016*.

Association for Computing Machinery, Inc, 57–80.

<https://doi.org/10.1145/3024906.3024910>

Sirocchi, C., Pofantis, A. O., & Bogliolo, A. (2022). Investigating Participation Mechanisms in EU Code Week. *ArXiv Preprint ArXiv:2205.14740*.

Skiena, S. S., & Revilla, M. A. (2003). Programming challenges: The programming contest training manual. *Acm SIGACT News*, 34(3), 68–74.

Smith, K., & Flores, M. A. (2019). *Teacher educators as teachers and as researchers*. Taylor & Francis.

Snyder, L., Barnes, T., Garcia, D., Paul, J., & Simon, B. (2012). The first five computer science principles pilots: Summary and comparisons. *ACM Inroads*, 3(2), 54–57.

Society, I. C. (2014). *Software Engineering Competency Model (SWECOM)*. IEEE-CS.

Society, R. (2017). After the reboot: Computing education in UK schools. *Policy Report*.

South Australia, U. (2020). *Resource on Academic Integrity*.

<https://lo.unisa.edu.au/course/view.php?id=6751§ion=6>

Stupnisky, R. H., BrckaLorenz, A., Yuhas, B., & Guay, F. (2018). Faculty members' motivation for teaching and best practices: Testing a model based on self-determination theory across institution types. *Contemporary Educational Psychology*, 53, 15–26.

Suarez-Alvarez, J. (2021). *Are 15-year-olds prepared to deal with fake news and misinformation?*

Sung, H.-Y., & Hwang, G.-J. (2013). A collaborative game-based learning approach to improving students' learning performance in science courses. *Comput Educ*, 63, 43–51.

Swartz, A. (2002). Musicbrainz: A semantic web service. *IEEE Intelligent Systems*, 17(1), 76–77.

Tasiopoulou, E., i, E. M., Gori, J. N., Xenofontos, N., Chovardas, A., Cinganotto, L., Anichini, G., Garista, P., Benedetti, F., Guida, M., Minichini, C., Benassi, A., &

- GrasVelazquez, A. (2020). *STE(A)MIT Integrated STEM teaching State of Play*. European Schoolnet.
- Taylor, C., & Sakharkar, S. (2019). '); DROP TABLE textbooks;– An Argument for SQL Injection Coverage in Database Textbooks. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 191–197.
- Taylor, C., Spacco, J., Bunde, D. P., Butler, Z., Bort, H., Hovey, C. L., & Zeume, T. (2018). Propagating the adoption of CS educational innovations. *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*, 217–235.
- Tedre, M., & Denning, P. J. (n.d.). The long quest for computational thinking. *ACM International Conference Proceeding Series. Association for Computing Machinery*, 2016, 120–129. <https://doi.org/10.1145/2999541.2999542>
- Tessler, J., Beth, B., & Lin, C. (2013). Using cargo-bot to provide contextualized learning of recursion. *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research*, 161–168.
- Thernstrom, A. T. (2009). *MCTS Self-Paced Training Kit (Exam 70-433): Microsoft® SQL Server® 2008—Database Development*. O'Reilly Media, Inc.
- T.K.I. Ministry of Education. (2017). *Technology in the New Zealand Curriculum*. <http://technology.tki.org.nz/Technology-in-the-NZC>
- Trautwein, U., Lüdtke, O., Köller, O., & Baumert, J. (2006). Self-esteem, academic self-concept, and achievement: How the learning environment moderates the dynamics of self-concept. *Journal of Personality and Social Psychology*, 90(2), 334.
- Trilling, B. (2014). Project management for learning: A foundational guide to applying project management principles and methods to education. *Newton Square: Project Management Institute Educational Foundation*.

- Trilling, R. (2018). Creating a new academic discipline: Cybersecurity management education. *Proceedings of the 19th Annual SIG Conference on Information Technology Education*, 78–83.
- Tsarava, K., Leifheit, L., & Ninaus, M. (2019). Cognitive correlates of computational thinking: Evaluation of a blended unplugged/Plugged-in course. *ACM International Conference Proceeding Series. Association for Computing Machinery*.
<https://doi.org/10.1145/3361721.3361729>
- Twarek, B., Freeman, C., Friend, M., Israel, M., Mora, L., & Ray, M. (2021). Infusing Equity and Inclusion in K-12 Computer Science Teacher Development. *2021 Conference on Research in Equitable and Sustained Participation in Engineering, Computing, and Technology (RESPECT)*, 1–2.
- UNESCO. (2017). UNESCO moving forward the 2030 Agenda for Sustainable Development. *United Nations Educ. Sci. Cult. Organ.*, 22.
- United Nations. (2022). *The sustainable Development Goals Report 2022*.
- United Nations. (2015). *THE 17 GOALS | Sustainable Development*. <https://sdgs.un.org/goals>
- University of Washington. (2022). *DO-IT. Disabilities, Opportunities, Internetworking, and Technology*. <https://www.washington.edu/doi/>
- Upadhyaya, B., McGill, M. M., & Decker, A. (2020). A longitudinal analysis of k-12 computing education research in the united states: Implications and recommendations for change. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE. Association for Computing Machinery*, 605–611.
<https://doi.org/10.1145/3328778.3366809>
- Upton, E., Duntemann, J., Roberts, R., Mamtora, T., & Everard, B. (2016). *Learning computer architecture with Raspberry Pi*. John Wiley & Sons.
- Urquiza-Fuentes, J., & Velázquez-Iturbide, J. Á. (2009). A survey of successful evaluations of program visualization and algorithm animation systems. *ACM Transactions on Computing Education (TOCE)*, 9(2), 1–21.

- Uzzell, D., Pol, E., & Badenas, D. (2002). Place identification, social cohesion, and environmental sustainability. *Environment and Behavior*, 34(1), 26–53.
- Vaccarezza, M. S., Kristjánsson, K., & Croce, M. (2023). *Phronesis (Practical Wisdom) as a Key to Moral Decision-Making: Comparing Two Models*.
- Vahrenhold, J., Nardelli, E., Pereira, C., & Berry, G. (2017). *Informatics education in Europe: Are we all in the same boat*.
- Viescas, J. L., & Hernandez, M. J. (2014). *SQL Queries for Mere Mortals: A hands-on guide to data manipulation in SQL*. Pearson Education.
- Vincenti, G., & Pecher, W. T. (2020). Merging Sustainability and Technology in the Classroom: An Experience Report. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 448–453.
- Vivian, R., & Falkner, K. (2018). A survey of Australian teachers' self-efficacy and assessment approaches for the K-12 digital technologies curriculum. *Proceedings of the 13th Workshop in Primary and Secondary Computing Education*, 1–10.
- Vivian, R., Quille, K., McGill, M. M., Falkner, K., Sentance, S., Barksdale, S., Busuttil, L., Cole, E., Liebe, C., & Maiorana, F. (2020). An international pilot study of k-12 teachers' computer science self-esteem. *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*, 117–123.
- Vuorikari, R., Gilleran, A., & Scimeca, S. (2011). Growing beyond innovators—ICT-based school collaboration in eTwinning. *European Conference on Technology Enhanced Learning*, 537–542.
- Waguespack, L., & Babb, J. (2019). Toward visualizing computing curricula: The challenge of competency. *Information Systems Education Journal*, 17(4), 51.
- Waguespack, L., Topi, H., Frezza, S., Babb, J., Marshall, L., Takada, S., van der Veer, G., & Pears, A. (2019). Adopting Competency Mindful of Professionalism in Baccalaureate Computing Curricula. *Proceedings of the EDSIG Conference ISSN, 2473*, 3857.

- Watts, P., & Kristjánsson, K. (2022a). Character education. In *Handbook of Philosophy of Education* (pp. 172–184). Routledge.
- Watts, P., & Kristjánsson, K. (2022b). Character education. In *Handbook of Philosophy of Education* (pp. 172–184). Routledge.
- Wendling, L., & Dumitru, A. (2021). *Evaluating the impact of nature-based solutions: A handbook for practitioners*. Directorate-General for Research and Innovation (European Commission).
- Wilcox, D., Thall, J., & Griffin, O. (2016). One canvas, two audiences: How faculty and students use a newly adopted learning management system. *Society for Information Technology & Teacher Education International Conference*, 1163–1168.
- Wildemuth, B. M., Yang, M., Hughes, A., Gruss, R., Geisler, G., & Marchionini, G. (2003). Access via features versus access via transcripts: User performance and satisfaction. In *TREC VID 2003 Noteb Pap.*
- Winch, R. F. (1965). Rosenberg: SOCIETY AND THE ADOLESCENT SELF-IMAGE (Book Review). *Social Forces*, 44(2), 255.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.
- Wing, J. M. (2020). *Computational thinking, 10 years later—Microsoft Research*.
<https://www.microsoft.com/en-us/research/blog/computational-thinking-10-years-later/>.
- Wobbrock, J. O., Kane, S. K., Gajos, K. Z., Harada, S., & Froehlich, J. (2011). Ability-based design: Concept, principles and examples. *ACM Trans Access Comput*, 3(3), 1–27.
- Wong, S. (2007). *StarUML Tutorial*.
- Woodward, B. S., Sendall, P., & Ceccucci, W. (2010). Integrating Soft Skill Competencies through Project-Based Learning across the Information Systems Curriculum. *Information Systems Education Journal*, 8(8), 8.

- Wu, B., Hu, Y., Ruis, A. R., & Wang, M. (2019). Analysing computational thinking in collaborative programming: A quantitative ethnography approach. *J Comput Assist Learn, 35*(3), 421–434. <https://doi.org/10.1111/jcal.12348>
- Xia, L., & Zhong, B. (2018). A systematic review on teaching and learning robotics content knowledge in K-12. *Computers & Education, 127*, 267–282.
- Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2016). Expanding computer science education in schools: Understanding teacher experiences and challenges. *Taylor Fr.* <https://www.tandfonline.com/doi/abs/10.1080/08993408.2016.1257418>.
- Yamamoto, K. K., Stodden, R. A., & Folk, E. D. R. (2014). Inclusive postsecondary education: Reimagining the transition trajectories of vocational rehabilitation clients with intellectual disabilities. *Journal of Vocational Rehabilitation, 40*(1), 59–71. <https://doi.org/10.3233/JVR-130662>
- Yehezkel, C., Yurcik, W., Pearson, M., & Armstrong, D. (2001). Three simulator tools for teaching computer architecture: Little Man computer, and RTLsim. *Journal on Educational Resources in Computing (JERIC), 1*(4), 60–80.
- Zhong, B., Wang, Q., Chen, J., & Li, Y. (2016). An Exploration of Three-Dimensional Integrated Assessment for Computational Thinking. *J Educ Comput Res, 53*(4), 562–590. <https://doi.org/10.1177/0735633115608444>
- Zhou, S., Livingston, I. J., Schiefsky, M., Shieber, S. M., & Gajos, K. Z. (2016). Ingenium: Engaging novice students with Latin grammar. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, 2016*, 944–956.
- Zingaro, D., & Porter, L. (2014a). Peer instruction in computing: The value of instructor intervention. *Computers & Education, 71*, 87–96.
- Zingaro, D., & Porter, L. (2014b). Peer instruction in computing: The value of instructor intervention. *Computers & Education, 71*, 87–96.