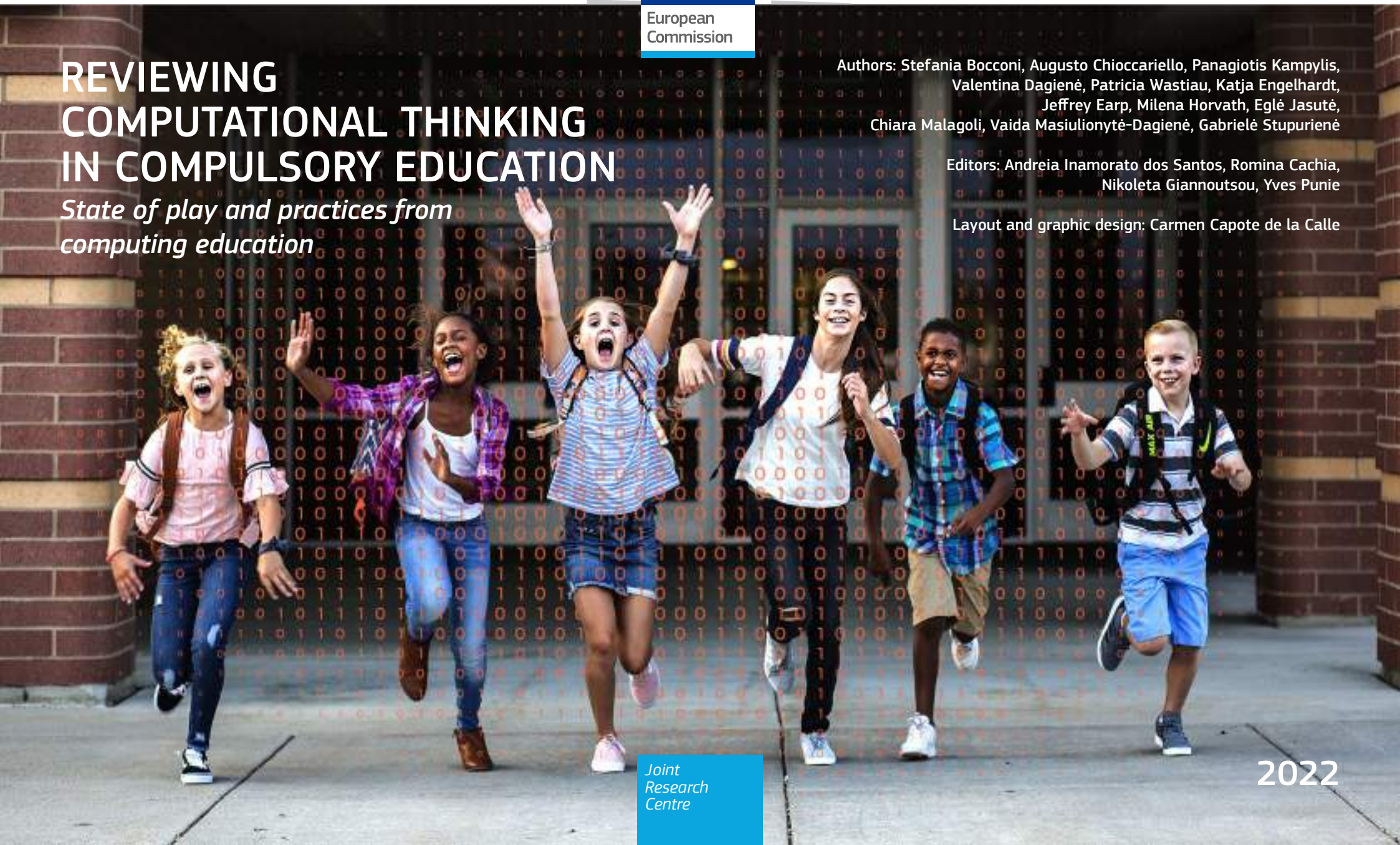


European Commission

REVIEWING COMPUTATIONAL THINKING IN COMPULSORY EDUCATION

State of play and practices from computing education

Authors: Stefania Bocconi, Augusto Chioccariello, Panagiotis Kampylis, Valentina Dagienė, Patricia Wastiau, Katja Engelhardt, Jeffrey Earp, Milena Horvath, Eglė Jasutė, Chiara Malagoli, Vaida Masiulionytė-Dagienė, Gabrielė Stupurienė

Editors: Andreia Inamorato dos Santos, Romina Cachia, Nikoleta Giannoutsou, Yves Punie

Layout and graphic design: Carmen Capote de la Calle

Joint Research Centre

2022

Reviewing Computational Thinking in Compulsory Education

State of play and practices from computing education

Contact information
Name: Yves Punie
Address: Edificio Expo, C/ Inca Garcilaso 3, E-41092 Seville (Spain)
Email: Yves.PUNIE@ec.europa.eu
Tel: +34 954 488 229

EU Science Hub
https://joint-research-centre.ec.europa.eu/

JRC128347

PDF
ISBN 978-92-76-47208-7
doi:10.2760/126955

Luxembourg: Publications Office of the European Union, 2022

Bocconi, S., Chioccariello, A., Kampylis, P., Dagienė, V., Wastiau, P., Engelhardt, K., Earp, J., Horvath, M.A., Jasutė, E., Malagoli, C., Masiulionytė-Dagienė, V. and Stupurienė, G., Reviewing Computational Thinking in Compulsory Education, Inamorato dos Santos, A., Cachia, R., Giannoutsou, N. and Punie, Y. editor(s), Publications Office of the European Union, Luxembourg, 2022, ISBN 978-92-76-47208-7, doi:10.2760/126955, JRC128347.

Abstract

In recent years, many European countries have revised their statutory compulsory education curriculum, introducing basic Computer Science concepts. This has paved the way for the development of students' Computational Thinking (CT) skills. Further impetus in this direction is coming from the European Commission's Digital Education Action Plan 2021-2027, where quality Computing Education is a key element under the priority "Enhancing digital skills and competences for the digital transformation". Despite increasing uptake, a range of issues and challenges are emerging for the effective integration of CT skills in compulsory education. This report updates and extends findings from the 2016 CompuThink study, providing an updated overview in 22 EU Member States and eight non-EU countries. The study has gathered a wide range of evidence from a systematic literature review, a survey with representatives of Ministries of Education, two online consultation events, and through in-depth case studies in nine European countries involving semi-structured interviews (with experts, policy makers, school leaders, teachers) and focus groups (with students). The report discusses significant developments concerning the integration of CT skills in compulsory education in Europe between 2016 and 2021. It also provides a comprehensive summary of evidence, including eleven recommendations for policy and practice.

Table of contents

Foreword

Computer Science education introduced from early age is regarded as a key enabler for the digital transformation of our society and economy. Its need has been made more evident during the COVID-19 pandemic. To this end, the teaching of Computer Science concepts in formal school education has increasingly been supported and implemented by the Ministries of Education of European countries, as well as internationally. The European Commission acknowledges this and reinforces the move in this direction via the Digital Education Action Plan (2021-2027), which presents computing education as one of the priorities to enhance digital skills and competences for the digital transformation. The integration of Computer Science concepts into the curriculum is relatively new, and it is therefore necessary to understand its challenges in practice in order to put in place provisions at policy level as well as other decision-making spheres.

It is with great pleasure that the Computational Thinking Study II is now published by the Joint Research Centre (JRC), a work done in collaboration with DG Education and Culture and with external experts. This report is a follow-up study to the first CompuThink report, published in 2016. It gathers evidence from 22 EU Member States, as well as eight non-EU countries, providing a comprehensive discussion of significant developments regarding the integration of CT skills into formal compulsory education curricula between 2016 and 2021. The findings of this report will also feed into two upcoming Council Recommendations, one on enabling factors for digital education and another on improving the provision of digital skills in education and training.

This report is part of the JRC research on "Learning and Skills for the Digital Era". Other key contributions to facilitate the digital transformation of education and training and the acquisition of digital skills comprise the Digital Competence Framework for Citizen (DigComp), the Digital Competence framework for Educators (DigCompEdu) and the digital competence self-reflection tools for teachers (SELFIEforTEACHERS) and for schools (SELFIE). In addition, there are the key competence frameworks for Personal, Social and Learning to Learn competences (LifeComp), for entrepreneurial competences (EntreComp) and for sustainability competences (GreenComp). More information on all our studies can be found on the JRC Science Hub.

Yves Punie
Acting Head of Unit
JRC Human Capital and Employment Unit
European Commission

Acknowledgements

The authors of this report would like to thank research officers from the European Commission's Joint Research Centre (JRC) Andreia Inamorato, Nikoleta Giannoutsou, Romina Cachia and Yves Punie, and policy officers from the Directorate-General Education and Culture Anusca Ferrari and Veronica Mobilio for their ongoing support throughout the research process and for their helpful feedback on an early draft of this report.

We are grateful to Simon Peyton Jones (UK), Morten Søby (NO), and Maciej Sysło (PL), who provided valuable and detailed feedback on the final draft of this report. Special thanks go to the participants in the online expert workshop held on 9 June 2021. Their insights have helped to steer the direction of this research and bring the final study results into sharper focus. Special thanks also go to the participants in the validation workshop held on 21 October 2021, who provided valuable insights on the preliminary findings of the study and the draft policy recommendations. The participants in both these online workshops are listed in Annex 2.

Finally, we wish to thank the policymakers, experts, school leaders, teachers, and students we interviewed for the nine in-depth case studies (see Annex 3). We gratefully acknowledge the schools who contributed to the study: Armfelt School (FI), Collège Vauban de Belfort (FR), College Alberto Giacometti Montigny Le Bretonneux (FR), Popovača Primary School (HR), Klaipėdos Gedminų progimnazija (LT), Knappskog School (NO), Samorządowa Szkoła Podstawowa nr 6 (PL), Vendelsomalmsskolan (SE), The Ludovit Stur Primary School (SK), and Harrogate Grammar School (UK-ENG). The case studies would not have been possible without their collaboration, sharing of experience and valuable feedback on the case study reports. A special mention goes to the three experts who provided input for the multiple-case studies: Ivan Kalaš (SK), Celia Hoyles (UK) and Peter Hubwieser (DE). Special thanks also go to the 28 Ministries of Education and other educational institutions who contributed to the survey (see Annex 1), as well as to the four experts who provided input on initial Vocational Education and Training in compulsory education settings (see Annex 3). Finally, we wish to thank Carmen Capote for overseeing the graphic design and layout of the report.

Authors

Stefania Bocconi

Augusto Chioccariello

Panagiotis Kampylis

Valentina Dagienė

Patricia Wastiau

Katja Engelhardt

Jeffrey Earp

Milena Horvath

Eglė Jasutė

Chiara Malagoli

Vaida Masiulionytė-Dagienė

Gabrielė Stupurienė

Executive Summary

In recent years, Computational Thinking (CT) has emerged as a fundamental skill for everyone, not just for computer scientists. More and more countries are promoting the development of Computational Thinking skills by introducing Computer Science into school curricula. However, questions remain regarding its implementation, specifically what Computer Science concepts should be taught, when and how. It should be noted that throughout this study, the terms *Computing*, *Computer Science* and *Informatics* are used interchangeably, in line with the European Commission's Digital Education Action Plan (DEAP) 2021-2027.

Many European Union (EU) Member States have recently revised their statutory curriculum by introducing basic Computer Science concepts into primary and lower secondary education for developing students' CT skills. To capture and analyse the unfolding process, this study has gathered a wide range of evidence from a systematic literature review, a survey with representatives of Ministries of Education (MoE), multiple-case studies (including semi-structured interviews and focus groups), and two online consultation events. This evidence primarily centres on implementation in **compulsory education**, which is the primary focus of this study.

In terms of coverage, the study collected data from **30 different countries**. The MoE survey was proposed to 34 European countries plus Singapore, with 28 (including Singapore) replying. This was complemented by input from the in-depth case studies, which were conducted with nine European countries (FI, FR, HR, LT, NO, PL, SE, SK, UK-ENG); seven of these had also replied to the MoE survey, Sweden and UK-England being the exceptions. So, all together, this study collected evidence from 22 EU Member States (AT, BE, CY, CZ, DK, EL, ES, FI, FR, HR, HU, IE, IT, LT, LU, MT, PL, PT, RO, SE, SI, SK), seven other European countries (CH, GE, IL, NO, RS, RU, UK-ENG), and Singapore.

The study concentrated on some key questions concerning the integration of CT skills in compulsory education:

- What is the state of play on integrating CT in EU compulsory education settings?
- What are the core characteristics of CT and its relationship with computer science, informatics and computing?
- How are CT skills being developed and assessed in EU compulsory education?
- How can computing education in the EU be improved?

Policy context

In the context of the **Digital Education Action Plan 2021-2027**, computing education is one of the requirements defined for the priority "Enhancing digital skills and competences for the digital transformation". The aforementioned plan points to the need to provide high-quality computing/Information Technology education to all students in Europe: "Computing education [also known as informatics or computer science in many countries] in schools allows young people to gain a sound understanding of the digital world. Introducing pupils to computing from an early age, through innovative and motivating approaches to teaching, in formal and non-formal settings, can help develop problem-solving, creativity, and collaboration skills. [...] A solid and scientific understanding of the digital world can build on, and complement, broader digital skills development."

There is a growing understanding that **digital competence goes beyond basic digital skills**, so there is a need to understand better how CT skills contribute to young people's skillsets and competences essential for the digital world we live in. This study links with other Joint Research Centre (JRC) studies on digital competence for citizens (DigComp and DigCompSAT), teachers (DigCompEdu and SELFIEforTEACHERS), and the digital capacity of schools (DigCompOrg and SELFIE).[1]

European Commission President Ursula von der Leyen announced in her 2021 State of the Union address that digital education and skills "need leaders' attention and a structured dialogue at top-level". This was echoed by Member State leaders in the October 2021 European Council conclusions, which underlined the need to focus on digital skills and education. Responding to this call, in October 2021 a project group of nine Commissioners announced the launch of the **Structured Dialogue with Member States on digital education and skills**. The Structured Dialogue delivers on Action 1 of the Digital Education Action Plan 2021-2027 and complements it by including digital skills in its scope. The dialogue invites Member States to agree jointly on the key enabling factors to make digital education and training effective and inclusive. It will include different branches and institutions of government, from education and training institutions to infrastructure providers, the private sector, social partners and civil society. The structured dialogue will run until the end of 2022. Based on its outcomes, the Commission will prepare proposals for two Council Recommendations by the end of the same year: on the enabling factors for digital education and on improving the provision of digital skills in education and training.

Key conclusions

Looking at the situation in the 29 European countries analysed for this report, 18 EU Member States (AT, BE, CY, EL, ES, FI, FR, HR, HU, IE, LT, LU, MT, PL, PT, RO, SE, SK) and seven other European countries (CH, GE, IL, NO, RS, RU, UK-ENG) have already introduced basic computer science concepts to some degree in their statutory curriculum for developing CT skills. In addition, one Member State (DK) is extensively piloting actions of this kind, while a further three are planning policies in this direction (CZ, IT, SI). Out of the 29 analysed European countries, 12 Member States (AT, EL, FI, FR, HU, LT, LU, MT, PL, PT, SE, SK) and five other European countries (CH, NO, RS, RU, UK-ENG) have introduced basic computer science concepts as compulsory for study in both in primary and lower secondary schools.

The positioning of basic Computer Science (CS) concepts in statutory curricula to develop CT skills varies from country to country. Three different approaches to integration are adopted:

- as a **cross-curricular theme** – basic CS concepts are addressed in all subjects, and all teachers share responsibility for developing Computational Thinking skills;
- as **part of a separate subject** – basic CS concepts are taught in a computing-related subject (e.g., Informatics);
- **within other subjects** – basic CS concepts are integrated within some curriculum subjects (e.g., Maths and Technology).

At the **primary level**, a combination of these three approaches is commonly adopted. In five countries (FI, LU, PT, SE – and RU), CT skills are developed as part of a cross-curricular theme and within other subjects. In another five countries (EL, HR, LT, PL, SK), CT skills are part of a separate subject and are also addressed as a cross-curricular theme. Lastly, in Cyprus, CT skills are part of a separate subject and are also addressed within other subjects too. CT skills are addressed as a purely cross-curricular theme in three countries (AT, LU, MT). In primary education, teachers cover several subjects in their practice without necessarily having specialised subject-area expertise in each one. This situation favours integration in a manner where CT skills as a cross-curricular theme is combined with another of the approaches. At the **lower secondary level**, disciplinary specialisation becomes prominent. Here, in 16 countries, CT skills are primarily integrated as part of a separate computing subject (AT, EL, HR, HU, IE, LT, LU, MT, PL, RO, SK - and CH, IL, RS, RU, UK-ENG). In six countries, they are addressed within other subjects (FI, FR, PT, SE - and GE, NO).

Evidence collected from in-depth case studies involving nine European countries shows that basic CS concepts integrated into curricula centre around the relationship between **"Algorithms"** and **"Programming"**, addressed at different levels of age-appropriate complexity. This relationship embodies a broad view of CT skills developed through problem-solving activities that include the formulation and design of the solution (algorithms) and the implementation process (programming). These computing concepts provide foundations for developing students' CT skills and increasing their scientific understanding of the digital world.

Generally, effective **pedagogical approaches** adopted to develop CT skills involve working on real-life problems and encouraging students to create their own programs, applications, animations, videogames, and so on. Another key aspect is promoting the adoption of development cycles when programming. At the primary level, commonly adopted pedagogical approaches include playful learning, learning by doing, learning from mistakes, and working in small groups. Students are introduced to basic CS concepts via hands-on, playful activities with programmable robots and block-based visual programming environments. By controlling robots or constructing programs through a sequence of instructions, learners gradually move from being passive consumers of technology to being creators of digital objects. At the lower secondary level, approaches focus on fostering problem-solving and logical thinking skills, promoting student autonomy/agency through project-based learning, game-based approaches, pair-programming, and learning individually. The value of debugging as a strategy is exploited both at primary and lower secondary level to create a culture of learning-through-error, encouraging students to re-evaluate the significance of mistakes as a means for furthering their expertise.

Across primary and lower secondary levels, formative assessment of students' CT skills mainly relies on teacher observation of students while developing their projects and solving problems; quizzes (e.g., Bebras tasks), exercises and surveys

for assessing CT skills are also commonly adopted. At the lower secondary level, summative assessment of CT skills takes on a more significant role (e.g., via exams and e-portfolios), and the focus is more on student mastery of programming tasks and understanding of their proposed solutions.

However, these teaching and assessment approaches require pedagogical and content knowledge that is yet to be acquired by most teachers involved in the implementation of computing education at the compulsory level. Three main challenges have emerged across primary and lower secondary education: by far the most significant of these is teacher upskilling and support, followed by competition with other curriculum priorities, and adoption of suitable assessment methods. So key policy recommendations for quality computing education centre around these three crucial areas.

A strong effort in providing professional development is called for to **upskill teachers in content and related pedagogy**. This is particularly critical, given that many teachers do not have a background in computing education. Accordingly, a number of essential measures are required, such as (i) the provision of quality training that involves medium and long-term training interventions, enacted regularly; and (ii) qualitative methodological support for teachers on how to handle the progression between grade levels in teaching basic CS concepts in an age-appropriate way, with the application of tools suitable for that purpose. Teachers need to gain confidence with basic CS contents and appropriate, sound pedagogical approaches that suit the nature and requirements of their students. To this end, professional development should go hand-in-hand with the activation of support measures with a marked emphasis on collaborative peer-support actions among teachers, such as networking and the sharing of experiences and concrete examples. Another essential measure in this direction is access to suitable, high-quality learning materials provided by different sources like educational authorities, teachers, grassroots initiatives, and publishers. Forming and sustaining school hubs that connect schools up for mutual support can help raise and spread quality in computing education. Clearly, the steady provision of adequate funding is necessary to ensure sustained teacher upskilling and also to provide incentives for schools and teachers to engage in professional development. From a long-term perspective, efforts should also be devoted to the inclusion of basic computing within the pre-service education of school teachers.

Given that the integration of basic CS concepts in the curriculum is relatively novel and is therefore in **competition with more established curriculum priorities**, it is crucial to ensure that provisions for this are made at different decision-making levels of the education system. In the first instance, this implies making adequate space in national curricula to develop students' Computational Thinking skills, setting a minimum number of hours for the regular teaching of CS concepts. Schools require sustained financial support so they can release staff to attend professional development opportunities and have the digital infrastructure they need to run activities like programming and educational robotics. Finally, when CT skills are positioned as a cross-curricular theme, it is crucial to clearly formulate and assign responsibility to teachers in charge of implementing basic CS concepts within the teaching curriculum.

Dedicating more attention to defining clear strategies that help teachers in **formative and summative assessment** is essential for improving quality computing education and monitoring the development of students' CT skills. Detailed criteria for assessing CT skills should be defined, encompassing both students' ability to produce successful programming solutions and build up their CT skills. Adoption of effective formative assessment methods is required so that both students and teachers can gain timely feedback during class activities, contributing to quality computing education. Another crucial assessment measure

is for CT skills development to be integrated into the final exam at the end of lower secondary school, indicating the importance of computing education to all concerned. This summative assessment is also essential for overall monitoring of CT skills development as a foundational component of compulsory education.

Computational Thinking is more than the promising new trend it was back in 2016. Computer Science concepts underpinning CT skills development have been steadily integrated as part of primary and lower secondary curricula across Europe. This integration is a clear sign that Ministries of Education are addressing the need to provide students with a scientific grounding to understand and operate in the digital world. As this process continues to evolve, monitoring and evaluation of the implementation of CS-related curricula will become crucial for collecting evidence on the effectiveness of adopted integration approaches.

1. Relevant JRC references:
- DigComp: https://joint-research-centre.ec.europa.eu/digcomp_en
- DigCompSAT: https://publications.jrc.ec.europa.eu/repository/handle/JRC123226
- DigCompEdu: https://publications.jrc.ec.europa.eu/repository/handle/JRC107466
- SELFIEforTEACHERS: https://educators-go-digital.jrc.ec.europa.eu/
- DigCompOrg: https://ec.europa.eu/jrc/en/digcomporg
- SELFIE: https://ec.europa.eu/education/schools-go-digital_en

Résumé analytique

Ces dernières années, la pensée informatique est apparue comme une compétence fondamentale pour tous, et pas seulement pour les informaticiens. De plus en plus de pays encouragent le développement de la pensée informatique en introduisant l'informatique dans les programmes scolaires. Cependant, des questions subsistent concernant plus précisément le choix des concepts qui devraient être enseignés, du moment et de la manière de les enseigner. Il convient de noter que tout au long de la présente étude, les termes informatique et science informatique sont utilisés de manière synonyme, conformément au Plan d'action en matière d'éducation numérique 2021-2027 de la Commission Européenne.

De nombreux États membres de l'Union Européenne (UE) ont récemment révisé leur programme d'enseignement officiels en introduisant des concepts informatiques de base dans l'enseignement primaire et secondaire inférieur afin de développer les compétences des élèves en matière de pensée informatique. Afin de saisir et d'analyser les développements en cours, cette étude rassemble des données provenant d'une revue systématique de la littérature, d'une enquête auprès de représentants des ministères en charge de l'éducation, d'une étude multi-cas (en ce inclus des entretiens semi-structurés et des groupes de discussion), et de deux séminaires en ligne de discussion entre experts. Ces données se focalise principalement sur la mise en œuvre au niveau l'**enseignement obligatoire**, qui est l'objet principal de cette étude.

En termes de couverture géographique, l'étude a recueilli des données provenant de **30 pays différents**. L'enquête du ministère de l'éducation a été proposée à 34 pays européens plus Singapour, dont 28 (y compris Singapour) ont répondu. Ces données ont été complétées par celles des études de cas approfondies, qui ont été menées auprès de neuf pays européens (FI, FR, HR, LT, NO, PL, SE, SK, UK-ENG); sept d'entre eux avaient également répondu à l'enquête des ministères de l'éducation, la Suède et le Royaume-Uni faisant exception. Au total, cette étude a donc recueilli des données auprès de 22 États membres de l'UE (AT, BE, CY, CZ, DK, EL, ES, FI, FR, HR, HU, IE, IT, LT, LU, MT, PL, PT, RO, SE, SI, SK), de sept autres pays européens (CH, GE, IL, NO, RS, RU, UK-ENG) et de Singapour.

L'étude s'est concentrée sur certaines questions clés concernant l'intégration des compétences en pensée informatique dans l'enseignement obligatoire:

- Quel est l'état d'avancement de l'intégration de la pensée informatique dans les établissements d'enseignement obligatoire de l'UE?
- Quelles sont les caractéristiques essentielles de la pensée informatique et quelle est sa relation avec l'informatique?
- Comment les compétences en pensée informatique sont-elles développées et évaluées dans l'enseignement obligatoire au sein de l'UE?
- Comment peut-on améliorer l'enseignement de l'informatique dans l'UE?

Contexte politique

Dans le cadre du **Plan d'action pour l'éducation numérique 2021-2027**, l'enseignement de l'informatique est l'une des exigences définies sous la priorité visant à "Renforcer les aptitudes et compétences numériques pour la transformation numérique". Le plan d'action sus-mentionné souligne la nécessité de dispenser un enseignement de haute qualité en informatique et technologies de l'information à tous les élèves en Europe: "L'enseignement de l'informatique (appelée également «science informatique» dans de nombreux pays) à l'école permet aux jeunes de bien comprendre le monde numérique. Le fait d'initier les apprenants à l'informatique dès leur plus jeune âge au moyen de méthodes d'enseignement innovantes et motivantes, tant dans des contextes formels qu'informels, peut les aider à acquérir des compétences en matière de résolution de problèmes, de créativité et de collaboration. [...] Une compréhension approfondie et scientifique du monde numérique peut être facilitée par le développement de compétences numériques plus générales et venir à son tour compléter ces compétences."

Il est devenu de plus en plus évident que **la compétence numérique va au-delà des compétences numériques de base**. Il est donc nécessaire de mieux comprendre comment les compétences en pensée informatique contribuent à l'ensemble des aptitudes et des compétences des jeunes qui sont essentielles dans le monde numérique dans lequel nous vivons. L'étude actuelle s'apparente aux autres études du JRC sur la compétence numérique des citoyens (DigComp et DigCompSAT), des enseignants (DigCompEdu et SELFIEforTEACHERS) et sur la capacité numérique des écoles (DigCompOrg et SELFIE).[2]

La présidente de la Commission Européenne, Ursula von der Leyen, a annoncé dans son discours sur l'état de l'Union de 2021 que l'éducation et les compétences numériques "nécessitent l'attention des dirigeants et un dialogue structuré au plus haut niveau". Les dirigeants des États membres s'en sont fait l'écho dans les conclusions du Conseil européen d'octobre 2021, qui ont souligné la nécessité de se concentrer sur les compétences et l'éducation numériques. En réponse à cet appel, un groupe de travail composé de neuf Commissaires a annoncé en octobre 2021 le lancement d'un **dialogue structuré avec les États membres sur l'éducation et les compétences numériques**. Ce dialogue structuré répond à l'action 1 du Plan d'action pour l'éducation numérique (2021-2027) et la complète en incluant les compétences numériques dans son champ d'application. Le dialogue invite les États Membres à convenir conjointement des facteurs clés permettant de rendre l'éducation et la formation numériques efficaces et inclusives. Il réunira différentes branches et institutions gouvernementales, des établissements d'enseignement et de formation aux fournisseurs d'infrastructures, en passant par le secteur privé, les partenaires sociaux et la société civile. Le dialogue structuré se déroulera jusqu'à la fin de 2022. Sur la base de ses résultats, la Commission proposera d'ici la fin de la même année des propositions pour deux recommandations du Conseil: l'une sur les facteurs favorables à l'éducation numérique, et l'autre sur l'amélioration de l'offre de compétences numériques dans l'éducation et la formation.

Principales conclusions

Parmi les 29 pays européens analysés, 18 États Membres de l'UE (AT, BE, CY, EL, ES, FI, FR, HR, HU, IE, LT, LU, MT, PL, PT, RO, SE, SK) et sept autres pays européens (CH, GE, IL, NO, RS, RU, UK-ENG) ont déjà introduit, jusqu'à un certain point, des notions de base en informatique dans leur programme d'enseignement officiel afin de développer des compétences en pensée informatique. En outre, un État

membre (DK) pilote à large echelle des actions de ce type, tandis que trois autres prévoient des politiques dans ce sens (CZ, IT, SI). Sur les 29 pays européens analysés, 12 États Membres de l'UE (AT, EL, FI, HU, LT, LU, MT, PL, PT, SE, SK) et cinq autres pays européens (CH, NO, RS, RU, UK-ENG) ont introduit les concepts de base de l'informatique comme matière obligatoire dans les écoles primaires et secondaires inférieures.

Le positionnement des concepts de base de l'informatique dans les curricula officiels pour développer les compétences en pensée informatique varie d'un pays à l'autre. Trois approches d'intégration différentes sont adoptées:

- en tant que **thème transversal** – les concepts fondamentaux des sciences informatiques sont abordés dans toutes les matières, et tous les enseignants partagent la responsabilité du développement des compétences en pensée informatique;
- en tant que **composante d'une matière distincte** – les concepts fondamentaux des sciences informatiques sont enseignés dans une matière liée à l'informatique;
- **dans le cadre d'autres matières** – les concepts fondamentaux des sciences informatiques sont intégrés dans d'autres matières du programme (par exemple, les mathématiques et la technologie).

Au niveau **primaire**, une combinaison de ces trois approches est couramment adoptée. Dans cinq pays (FI, LU, PT, SE – et RU), les compétences en pensée informatique sont développées dans le cadre d'un thème transversal et dans le cadre d'autres matières. Dans cinq autres pays (EL, HR, LT, PL, SK), les compétences en pensée informatique font partie d'une matière distincte et sont également abordées dans le cadre d'un thème transversal. Enfin, à Chypre, les compétences en pensée informatique font partie d'une matière distincte et sont également abordées dans le cadre d'autres matières. Les compétences en pensée informatique sont abordées en tant que thème purement transversal dans trois pays (AT, LU, MT). Dans l'enseignement primaire, les enseignants couvrent plusieurs matières dans leur pratique sans nécessairement avoir une expertise spécialisée dans chacune d'entre elles. Cette situation favorise l'intégration d'une manière telle que les compétences en pensée informatique en tant que thème transversal sont combinées avec une autre des approches. Au niveau du **secondaire inférieur**, la spécialisation disciplinaire devient prépondérante. Dans 16 pays, les compétences en pensée informatique sont principalement intégrées dans le cadre d'une matière informatique distincte (AT, EL, HR, HU, IE, LT, LU, MT, PL, RO, SK – et CH, IL, RS, RU, UK-ENG). Dans six pays, elles sont abordées dans le cadre d'autres matières (FI, FR, PT, SE – et GE, NO).

Les données recueillies dans le cadre des études multi-cas approfondies menées dans neuf pays européens montrent que les concepts informatiques de base intégrés dans les programmes scolaires sont centrés sur la relation entre "**algorithmes**" et "**programmation**", abordée à différents niveaux de complexité selon l'âge. Cette relation incarne une vision large des compétences en pensée informatique, qui sont développées par des activités de résolution de problèmes incluant la formulation et la conception d'une solution (algorithmes) et sa mise en œuvre (programmation). Ces concepts informatiques constituent les fondements du développement des compétences en pensée informatique des élèves et de leur compréhension scientifique du monde numérique.

En général, les **approches pédagogiques** efficaces adoptées pour développer les compétences en pensée informatique impliquent que les élèves travaillent sur des problèmes de la vie réelle et les encouragent à créer quelque chose par eux-mêmes, qu'il s'agisse de programmes, d'applications, d'animations, de jeux vidéo, etc. Un autre aspect essentiel est la promotion d'un processus itératif de développement de programmes. Au niveau primaire, les approches pédagogiques couramment mises en œuvre comprennent l'apprentissage ludique, l'apprentissage par la pratique, l'apprentissage par l'erreur et le travail en petits groupes. Les élèves sont initiés aux concepts informatiques de base par le biais d'activités pratiques et ludiques avec des robots programmables et des environnements de programmation visuelle basés sur des blocs. En contrôlant des robots ou en construisant des programmes à partir d'une séquence d'instructions, les apprenants passent progressivement du statut d'utilisateurs passifs de la technologie à celui de créateurs d'objets numériques. Au niveau secondaire

inférieur, les approches se concentrent sur l'encouragement des compétences en matière de résolution de problèmes et de raisonnement logique, sur la promotion de l'autonomie/initiative des élèves par le biais de l'apprentissage par projet, d'approches basées sur le jeu, de la programmation en binôme et d'approches d'apprentissage individualisé. La valeur du débogage est exploitée tant au niveau primaire qu'au secondaire inférieur pour créer une culture de l'apprentissage par l'erreur, en encourageant les élèves à réévaluer la signification des erreurs comme moyen de renforcer leur expertise.

Dans le primaire et le secondaire inférieur, l'évaluation formative des compétences des élèves en matière de pensée informatique se base principalement sur l'observation des élèves par les enseignants lors de l'élaboration de projets, la résolution de problèmes et l'utilisation de quiz (par exemple, les tâches Bebras), d'exercices et d'enquêtes. Au niveau du secondaire inférieur, l'évaluation sommative des compétences en pensée informatique (par exemple, les examens, le portefeuille électronique) joue un rôle plus important et se concentre sur la maîtrise des tâches de programmation par les élèves et la compréhension des solutions qu'ils proposent.

Ces approches d'enseignement et d'évaluation nécessitent toutefois des connaissances pédagogiques et de contenu qui n'ont pas encore été acquises par la majorité des enseignants impliqués dans la mise en œuvre de l'enseignement de l'informatique au niveau obligatoire. Dans l'ensemble de l'enseignement primaire et secondaire inférieur, trois grands défis sont apparus: le plus important, et de loin, est le perfectionnement et le soutien à apporter aux enseignants, suivi de la concurrence vis-à-vis d'autres priorités du programme scolaire, et l'adoption de méthodes d'évaluation appropriées. Les principales recommandations politiques pour un enseignement informatique de qualité s'articulent autour de ces trois dimensions/domaines.

Un effort important en matière de développement professionnel est nécessaire pour **améliorer les compétences des enseignants, tant au niveau du contenu que de la pédagogie**. Ceci est particulièrement important, étant donné que la plupart des enseignants n'ont pas de formation en informatique. Cela implique (i) la mise en place d'une formation de qualité qui prévoit des interventions de formation à moyen et long terme, mises en œuvre sur une base régulière ; et (ii) un soutien méthodologique qualitatif sur la façon de gérer la progression entre les niveaux scolaires en enseignant les concepts informatiques de base selon l'âge et en utilisant des outils appropriés. Les enseignants doivent se familiariser avec les contenus de base en informatique et avec les approches pédagogiques adaptées à leurs élèves. À cette fin, le développement professionnel devrait aller de pair avec l'activation de mesures de soutien, en mettant particulièrement l'accent sur les actions d'apprentissage et d'échanges entre enseignants, telles que la mise en réseau, le partage d'expériences et d'exemples concrets. L'accès à des supports d'apprentissage adaptés et de qualité fournis par différentes sources (par exemple, les autorités éducatives, les enseignants, les initiatives de terrain, les éditeurs). La création pérenne de pôles scolaires qui se connectent et se soutiennent mutuellement peut contribuer à améliorer la qualité globale de l'enseignement de l'informatique. Un financement durable est essentiel pour garantir et maintenir la formation continue des enseignants, ainsi que pour inciter les écoles et les enseignants à s'engager dans leur développement professionnel. Dans une perspective à long terme, des efforts devraient être déployés dès maintenant pour inclure l'informatique de base dans la formation initiale des enseignants de l'enseignement obligatoire.

Étant donné que l'intégration des concepts informatiques de base dans le programme scolaire est relativement récente et entre donc en **concurrence avec des priorités plus établies**, il est crucial de s'assurer que des dispositions sont prises aux différents niveaux de décision du système éducatif. En premier lieu, cela implique d'accorder une place adéquate dans le programme national pour le développement des compétences en pensée informatique des élèves, en fixant un nombre minimum d'heures pour enseigner les concepts informatiques de base de manière régulière. Afin d'assurer le perfectionnement des enseignants, les écoles devraient bénéficier d'un soutien financier durable pour permettre à leur personnel de participer à des activités de développement professionnel et pour garantir la disponibilité d'une infrastructure numérique adéquate pour les activités

de programmation et de robotique éducative. Enfin, lorsque les compétences en pensée informatique sont intégrées en tant que thème transversal, il convient de formuler et d'attribuer clairement aux enseignants la responsabilité d'intégrer les concepts informatiques de base dans le programme scolaire.

Il est essentiel de consacrer plus d'attention à la définition de stratégies claires pour aider les enseignants en matière d'**évaluation formative et sommative** afin d'améliorer la qualité de l'enseignement de l'informatique et de suivre le développement des compétences en pensée informatique des élèves. Il convient de définir des critères détaillés pour l'évaluation des compétences en pensée informatique, englobant à la fois la capacité des élèves à produire des solutions de programmation réussie et leurs compétences en pensée informatique. Des méthodes efficaces sont nécessaires pour l'évaluation formative, afin que les élèves et les enseignants puissent bénéficier d'un retour d'information en temps utile pendant les activités d'apprentissage, contribuant ainsi à un enseignement informatique de qualité. L'évaluation du développement des compétences en pensée informatique devrait être intégrée à l'examen final à la fin du premier cycle de l'enseignement secondaire, ce qui indiquerait clairement l'importance accordée à l'enseignement de l'informatique. Cette évaluation sommative est également essentielle pour suivre le développement des compétences en pensée informatique en tant que composante fondamentale de l'enseignement obligatoire.

La pensée informatique est plus que la nouvelle tendance prometteuse qu'elle était en 2016. Les concepts informatiques qui sous-tendent le développement des compétences en pensée informatique ont été intégrés de manière continue dans le cadre des programmes d'enseignement du primaire et du premier cycle du secondaire dans toute l'Europe. C'est un signe clair que les ministères de l'éducation répondent à la nécessité de fournir aux élèves des bases scientifiques pour comprendre et fonctionner dans un monde numérique. Au fur et à mesure que ce processus évolue, le suivi et l'évaluation de la mise en œuvre des programmes d'enseignement intégrant la pensée informatique deviendront cruciales pour recueillir des preuves de l'efficacité des approches d'intégration mises en œuvre.

2. Références pertinentes du JRC:
- DigComp: https://joint-research-centre.ec.europa.eu/digcomp_en
- DigCompSAT: https://publications.jrc.ec.europa.eu/repository/handle/JRC123226
- DigCompEdu: https://publications.jrc.ec.europa.eu/repository/handle/JRC107466
- SELFIEforTEACHERS: https://educators-go-digital.jrc.ec.europa.eu/
- DigCompOrg: https://ec.europa.eu/jrc/en/digcomporg
- SELFIE: https://ec.europa.eu/education/schools-go-digital_en

1

Introduction

One of the six European Commission priorities for 2019-2024 is "A Europe fit for the digital age", the EU's digital strategy to "empower people with a new generation of technologies". Digital transformation has affected all spheres of society and the economy, with an ever-deepening impact on everyday life. However, before the COVID-19 pandemic, the impact on education and training was much more limited. While the pandemic demonstrated the need to accelerate the digital transformation of education and training systems, it also led to the amplification of existing challenges and inequalities between those who have access and competence to grasp the benefits of digital technologies and those who do not (e.g., Blaskó et al., 2021; Cachia et al., 2021).

In this context, the 2021 European Commission´s "Digital Education Action Plan (2021-2027) – Resetting education and training for the digital age" (European Commission, 2020) is a renewed European Union (EU) policy initiative to support the sustainable and effective adaptation of education and training systems in EU Member States to make them more fit for the digital age. The Digital Education Action Plan has set two priority areas: 1) Fostering the development of a high-performing digital education ecosystem, and 2) enhancing digital skills and competences for digital transformation. Computing education is listed under the second priority area as one of the requirements for strengthening young people's digital skills and competences, which are essential "to gain a critical and practical understanding of the digital world in which they live".[3]

The research study "Reviewing Computational Thinking in Compulsory Education: State of Play and Practices from the Field" presented in this report was designed, funded and followed by the European Commission's Joint Research Centre (JRC) to investigate how Computational Thinking (CT) is currently positioned within compulsory school education in Europe's various Member States, as well as outside the EU. The study was carried out from April to December 2021 by the Institute for Educational Technology of the Italian National Research Council (CNR-ITD), together with European Schoolnet (EUN) and Vilnius University (VU). The study's aim was to update the earlier JRC report "Developing Computational Thinking in Compulsory Education: implications for policy and practice," produced for 2016 CompuThink study (Bocconi et al., 2016, which provided a comprehensive overview of research findings, as well as grassroots and policy initiatives for developing Computational Thinking skills in compulsory education in Europe.

3. https://education.ec.europa.eu/focus-topics/digital-education/digital-education-action-plan/action-10

Since the publication of the 2016 report much has changed, including the challenges and opportunities arising from the COVID-19 pandemic. In recent years, there has been a growing understanding that digital competence goes beyond basic digital skills. The Eurydice report on digital competence (European Commission/EACEA/Eurydice, 2019, p.10) states that "Half of the European education systems are currently reforming [their] curriculum related to digital competence. The revisions aim either at introducing digital competence into the curriculum where it had not previously been addressed, or making the subject area more prominent. Some reforms are also about changing the curriculum approach, updating content or strengthening particular areas such as coding, computational thinking or safety."

In this context, the study presented in this report contains a systematic review of the latest research findings, policy making, and grass-roots initiatives on the position of CT skills within Europe's compulsory education landscape. The study approaches the matter from theoretical, organisational, and practical perspectives in the endeavour to spotlight new understandings, developments and emerging trends, bringing them into sharper critical focus.

Box 1

Key terms adopted[4]

In this report, the following working terms are adopted.

Computing education

encompasses basic *Computer Science* concepts (i.e., algorithms and programming) for developing *Computational Thinking skills*.

Computer Science

is used interchangeably with *Computing* and *Informatics*, in line with the European Commission's Digital Education Action Plan 2021-2017 (p.13).

Computational Thinking skills

encompasses abstraction, algorithmic thinking, automation, decomposition, debugging and generalization (2016 EC Computational Thinking Study, Bocconi et al, 2016 p.18).

Computing curriculum

refers to a separate subject encompassing two main strands, i.e., basic *Computer Science* contents, and elements of *Digital Competence/Digital Literacy*.

4. A complete Terminology List is provided towards the end of this report.

1.1 The study's aims and objectives

This study aims to provide a comprehensive overview of the integration of CT skills in compulsory education in Europe. Its specific objectives are to shed light on and respond to four **research questions**, as depicted in Figure 1.

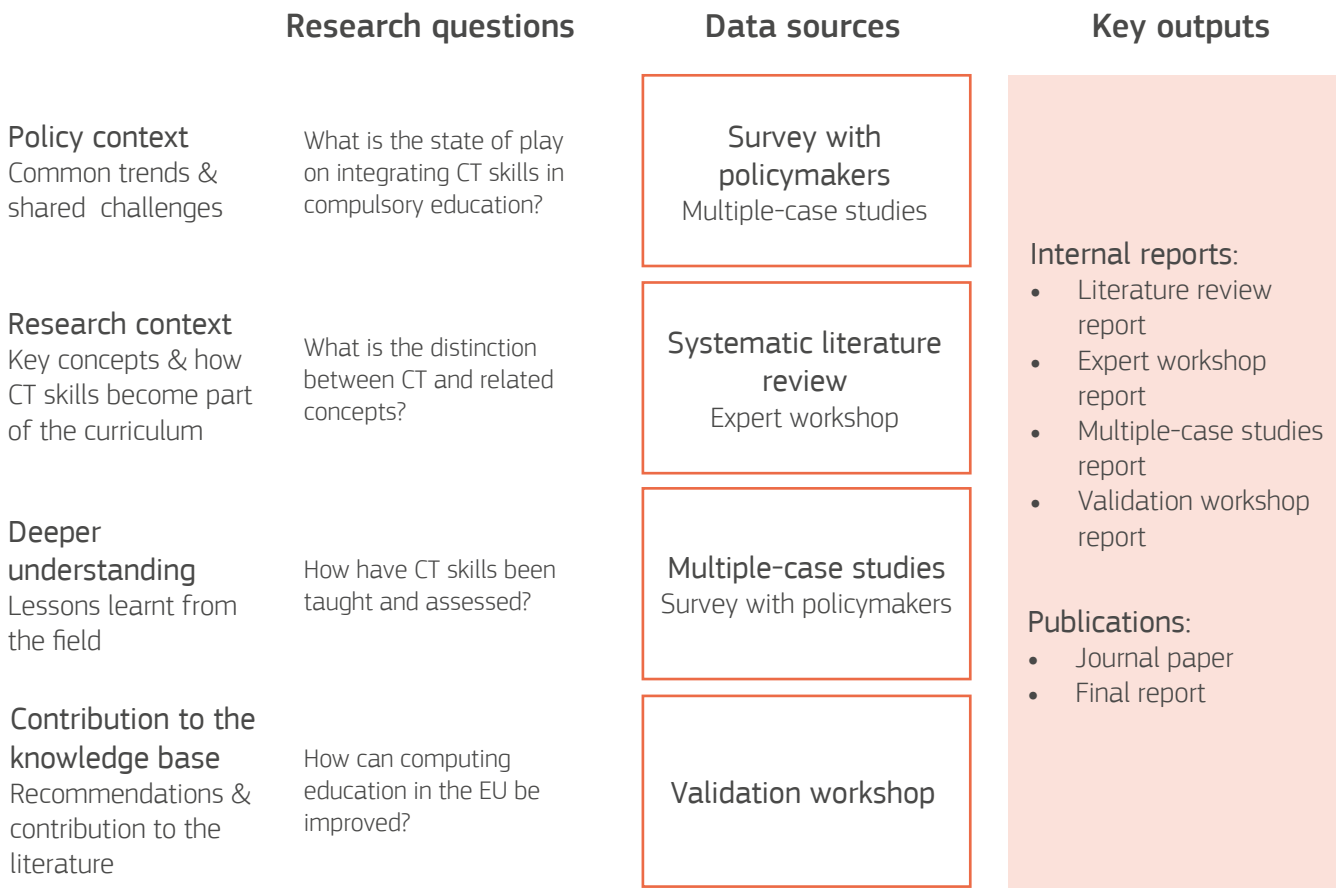


Figure 1. The study's context, research questions, data sources and key outputs
Source: Authors' elaboration

2

Research methodology

The study followed a qualitative approach to understand current research and practical understanding of CT skills in compulsory education, with a special focus on Europe. Within this approach different research methods were adopted; these are described in detail in the sections below, while a brief overview is provided in Figure 2.

Figure 2. Overall methodological approach and main components of the study
Source: Authors' elaboration

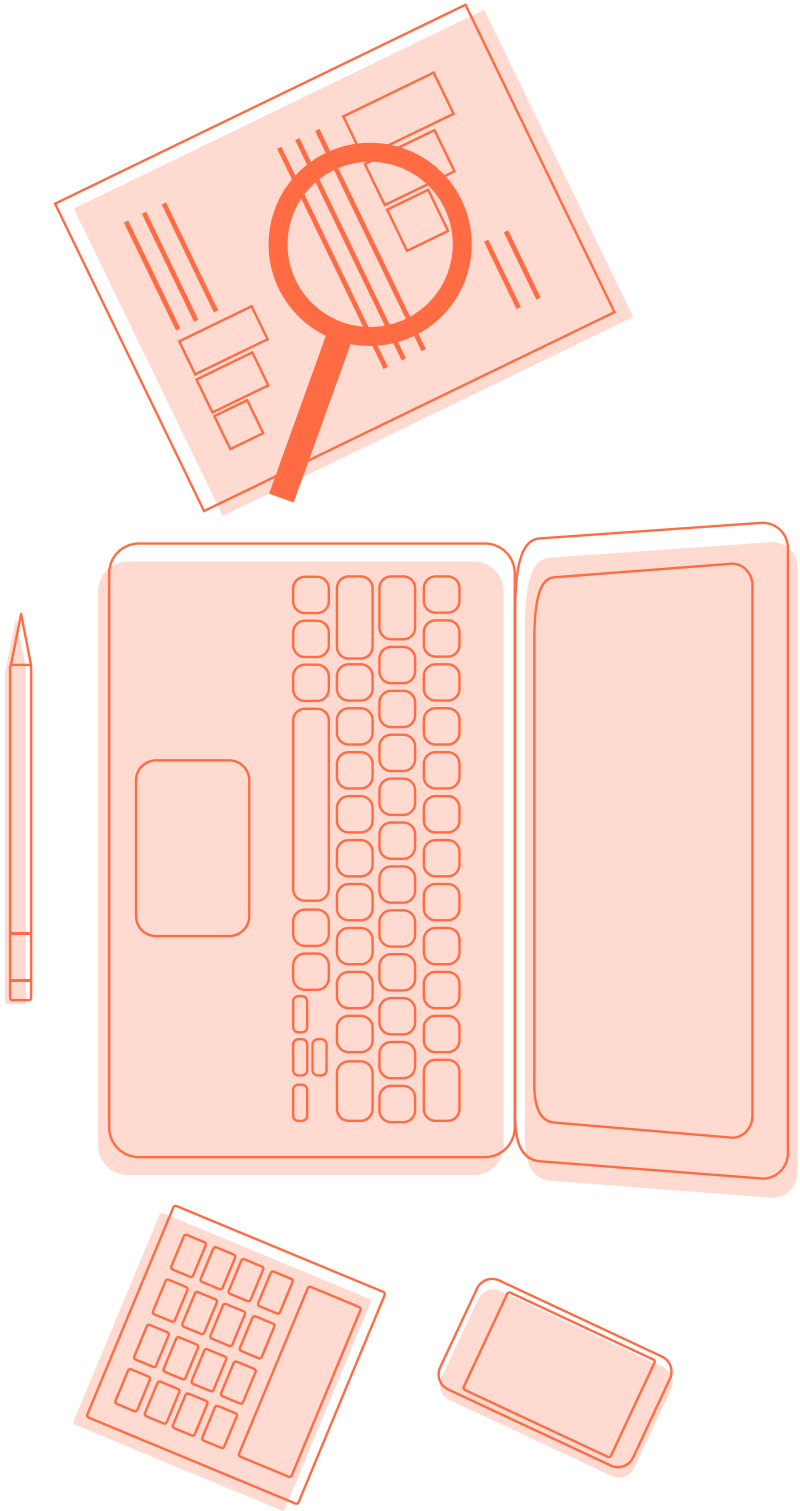
2.1 Desk research

A **systematic review of academic and grey literature** was carried out to reflect on what has changed in the CT field since 2016 when the report Developing Computational Thinking in Compulsory Education (Bocconi et al., 2016) was published. The commonly adopted steps of the PRISMA workflow, namely identification, screening, eligibility and inclusion, were applied (Moher et al., 2009).

The literature review **gathered 1869 publications** – 1143 academic works (peer-reviewed journal papers and book chapters) and 726 grey literature works (conference papers, reports, etc.). Screening the titles and abstracts of these 1869 using specific exclusion criteria resulted in a list of 478 publications for full-text analysis. This analysis yielded a core set of **98 significant publications analysed in-depth through a review matrix**. This set of significant publications comprises 53 academic and 46 grey literature works.

The review matrix specially devised for full-text analysis of these publications addressed various criteria, including the study's research questions. It allowed comparison of different sources, and the identification of significant findings, emerging patterns, and missing or inadequate elements that require further investigation in the subsequent phases of the study.

Following an **open data approach**, the set of publications gathered and analysed in the study are documented in a **Zotero open library**, to be hosted on the Zenodo[5] open research platform.



5. The open library is expected to be uploaded on Zenodo (https://zenodo.org) in Autumn 2022.

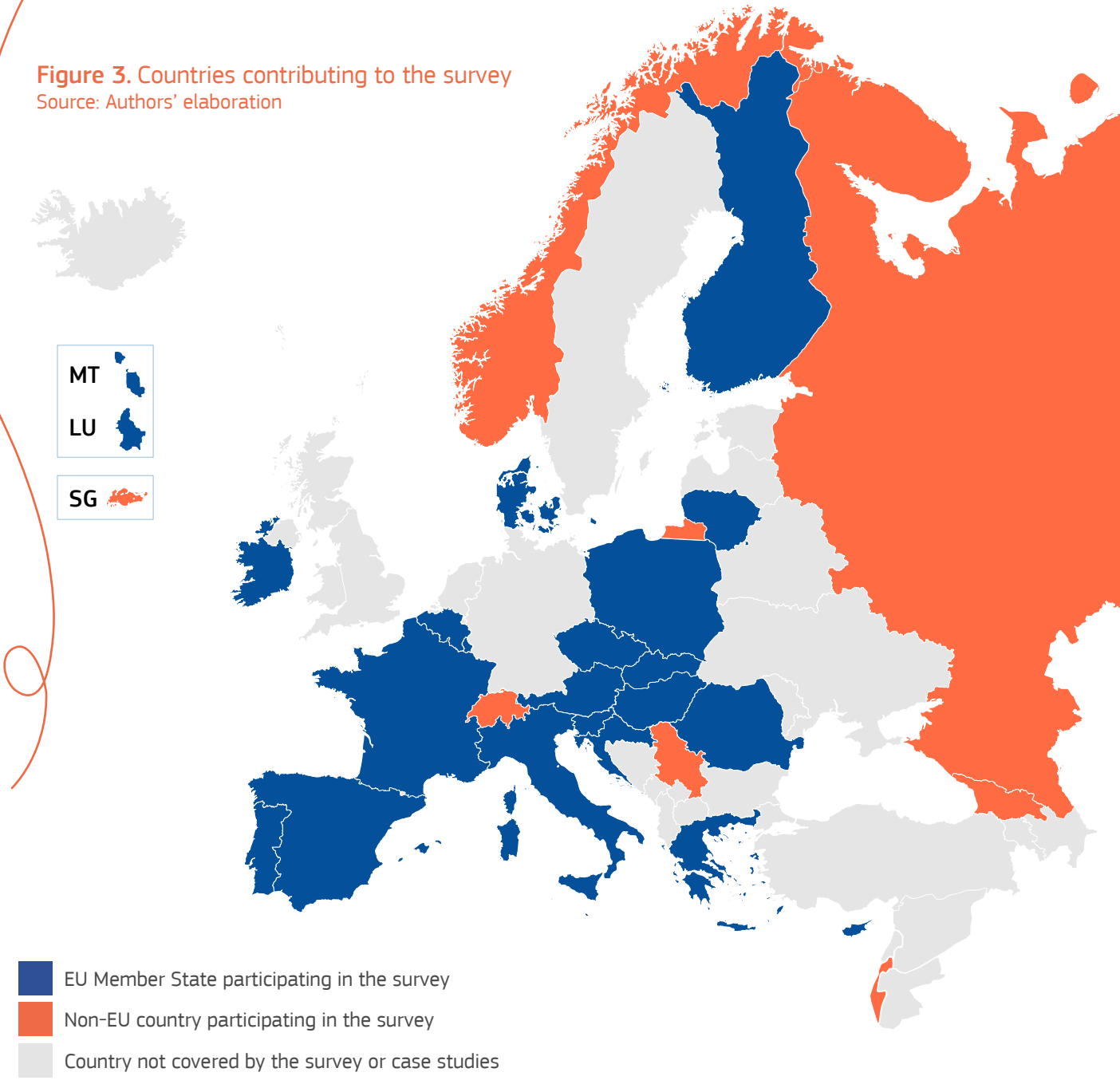
2.2 Survey of policy initiatives

The desk research was complemented by a survey comprising 44 open and closed items designed to gain an up-to-date picture of how Computational Thinking (and related concepts) are perceived, understood, and integrated into compulsory education in Europe and beyond. This survey was addressed to representatives from (or nominated by) 33 Ministries of Education within the European Schoolnet (EUN) umbrella, as well as Russia and Singapore. In total, representatives from Ministries of Education (or organisations nominated to act on their behalf) from **28 countries completed the survey** (including **21 EU Member States**), with responses varying in completeness and depth. These countries were:

Austria, Belgium Flanders & Belgium Wallonia, Croatia, Cyprus, Czech Republic, Denmark, Finland, France, Greece, Hungary, Ireland, Italy, Lithuania, Luxembourg, Malta, Poland, Portugal, Romania, Slovakia, Slovenia, Spain – and Georgia, Israel, Norway, Russia, Serbia, Singapore, Switzerland.[6]

This survey was also instrumental in **identifying and gaining access to more than 120 key documents**, such as curricula, guidelines, policy strategies and country reports. These documents were integrated into the desk research, while the insights gained into current ministerial priorities and work in progress supplemented the findings from the desk research and the multiple-case studies.

Figure 3. Countries contributing to the survey
Source: Authors' elaboration



6. Throughout this report, countries are listed in alphabetical order, both when full names and when two-letter codes are used. EU Member States are listed first, followed by the non-EU countries.

2.3 In-depth case studies & interviews

The data collected from the desk research and the survey was complemented by input from case studies and interviews. Altogether, **nine individual case studies** were carried out, and a **multiple-case study methodology** (Yin, 2014) was adopted to explore the integration of CT skills within the curriculum in different contexts. All the case studies selected for presentation in this report fall **within compulsory education in EU Member States**, given compulsory education has a broad mandate to provide all students with key competences.

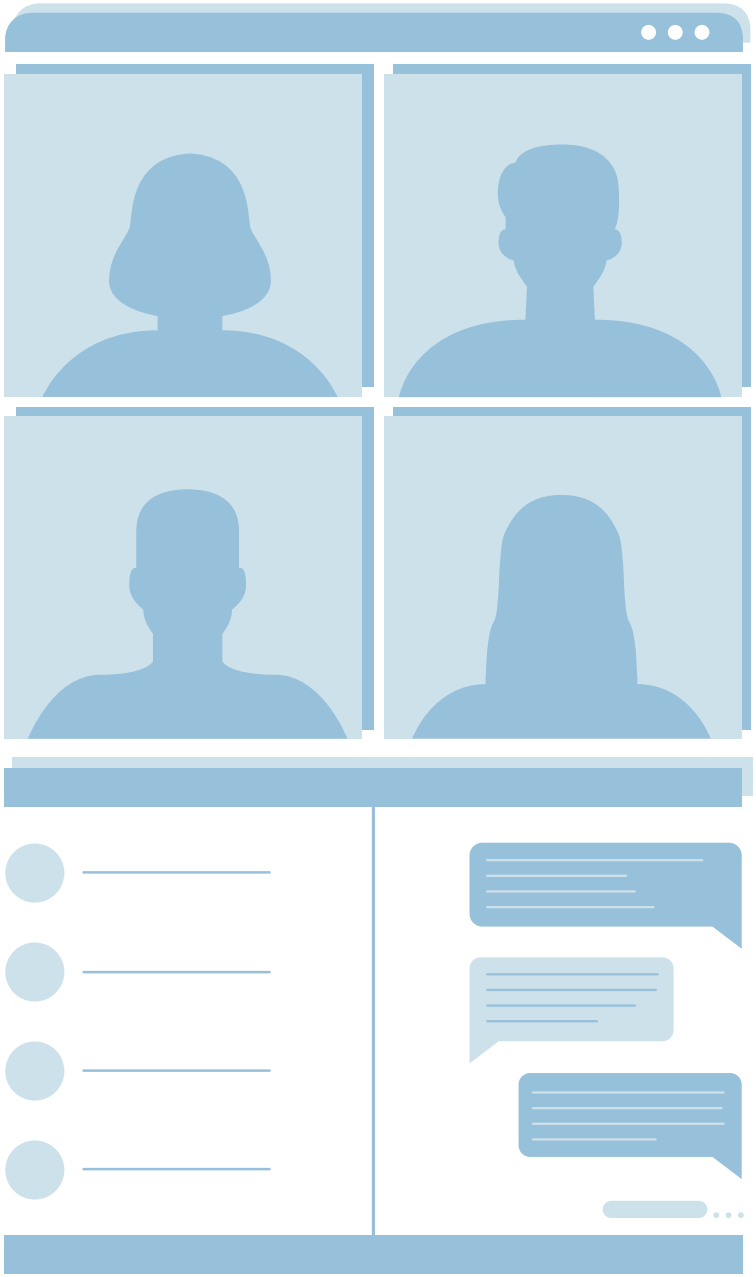
The **three Multiple-Case Studies (MCSs)** conducted cover each of the three CT integration approaches identified in the 2016 CompuThink study (Bocconi et al., 2016) and the survey of policy initiatives conducted for this study. These are presented in Table 1 below.

Each multiple-case study comprises three individual cases. Two cases within each MCS were selected as literal replication, i.e., likely to generate very similar or consolidated findings. The third case was treated as theoretical replication, i.e., as a "sounding board" that confirmed or contrasted with the previous results.

Replication strategy	MCS1: CT skills as a cross-curriculum theme at primary level	MCS2: CT skills as part of a separate subject at lower secondary level	MCS3: CT skills within other subjects at lower secondary level
Literal replication	C1 - Lithuania C2 - Norway	C4 - Croatia C5 - Poland	C7 - France C8 - Finland
Theoretical replication	C3 - Slovakia	C6 – UK-England	C9 - Sweden

Table 1. The structure and focus of the three multiple-case studies
Source: Authors' elaboration

In total, **88 subjects representing different stakeholder categories** (policymakers, experts, school leaders, teachers, and students) took part in the three multiple-case studies through **38 semi-structured interviews and ten focus groups**. As an additional data source, interviewees provided **documents related to integrating CT skills in the curriculum**. These included information on final exams, CT skills assessment means, references to platforms/tools used in classrooms, and CT-relevant learning materials teachers had produced for their classes. These documents informed the case studies and were added to the knowledge base gathered as part of the survey of policy initiatives.



2.4 Online consultations

We conducted two consultation events online, one at the outset and one at the end of the study. The first event, held on 9 June 2021, was an **online expert workshop** gathering 20 internationally-renowned scholars from 13 countries. They provided insights on how CT skills relate to Computer Science and Informatics, as well as on trends and challenges regarding the integration of CT skills into compulsory education curricula in Europe and beyond.

The second event was a **validation workshop** that took place on 21 October 2021. The objectives here were to discuss and validate the study's key outcomes and collect insights on how to improve computing education across the EU. Altogether, 37 education policymakers, stakeholders, and practitioners from 23 countries accepted the invitation to participate; they included representatives from each of the nine in-depth case studies. The combined list of participants in the expert and validation workshops reflects a balanced representation of backgrounds, genders, and nationalities. The names and affiliations of participants in both consultation events are presented in Annex 2.

3

Understanding Computational Thinking and related concepts

*"Computational thinking is the thought processes involved in **formulating a problem and expressing its solution(s)** in such a way that a computer — human or machine — can effectively carry out."*

(Wing, 2017)

3.1 How Computational Thinking is defined in the literature and in practice

Despite intensive analysis and investigation in the Computational Thinking (CT) field stretching back over the past fifteen years, researchers have not yet reached a consensus on a single 'banner' definition of CT. As a result, researchers tend to draw from a variety of proposed definitions when conducting their investigations. Accordingly, they assume different perspectives when applying, interpreting, and assessing CT concepts and related practices.

The European Commission's Staff Working Document accompanying the Digital Education Action Plan 2021-2027 (DEAP)[7] gives a glossary definition that describes Computational Thinking (including programming and coding) thus: "computational thinking, programming and coding are often used interchangeably in education settings, but they are distinct activities. Programming refers to the activity of analysing a problem, designing a solution and implementing it. Coding means implementing solutions in a particular programming language. Computational thinking, shorthand for "thinking as a «computer scientist» refers to the ability to understand the underlying notions and mechanisms of digital technologies to formulate and solve problems."

This definition reflects that proposed by Jeannette M. Wing (2017): "Computational thinking is the thought processes involved in formulating a problem and expressing its solution(s) in such a way that a computer – human or machine – can effectively carry out." This perspective is clearly rooted in the concepts and practices of Computer Science (i.e., thinking as a computer scientist) proposed as an intellectual framework for thinking. Wing's vision (2017) is that "computational thinking will be a fundamental skill – just like reading, writing, and arithmetic – used by everyone by the middle of the 21st Century." This proposal has been widely accepted as a basis for including Computational Thinking (CT) in K-12 education as a key competence for the 21st century. However, the debate over CT's definition and implications for education continues.

7. https://education.ec.europa.eu/sites/default/files/document-library-docs/deap-swd-sept2020_en.pdf

The range of different definitions found in the literature falls into three general areas:

- Computational Thinking can be understood as a way of thinking for developing solutions that can be processed and executed by a computational agent, namely a computer or robot – CT also embraces recognition of aspects of real-world problems suited to computational formulation (e.g., Corradini et al., 2017; Eickelmann et al., 2019);
- Computational Thinking is a way of thinking (thought process) for problem solving (Grover & Pea, 2018; Hazzan et al., 2020; Zhang & Nouri, 2019);
- Computational Thinking is defined as a thinking skill that can be transferred and applied in the process of solving real-world and significant problems in various contexts and disciplines through algorithmic methods (Israel-Fishelson et al., 2021; Román-González et al., 2019; Shute et al., 2017).

There is general acknowledgement that CT is not only a problem-solving process: the resulting solution to the problem must be expressed in a way that allows a computational agent to execute it.

Recent systematic literature reviews (Tikva & Tambouris, 2021) position definitions of CT in two broad categories: 1) **domain-specific**, i.e., problem-solving in computer science or programming; and 2) **domain-general**, namely systematic problem solving in everyday life, including in learning processes. Tang et al. (2020) identify competences applicable to both specific and general domains, while also pointing to the intrinsic programming and computing-related facets of CT; they affirm that many definitions are deeply intertwined with programming and computing.

Román-González, Moreno-León, & Robles (2017a) identify three interwoven threads, or macro-categories: 1) **generic definitions**, i.e., CT as a thought process that resonates with computing/programming disciplines, but can be independent of them; 2) **operational-model** definitions, which break down CT into sets of fundamental competences/practices, like abstraction and generalisation, that are firmly rooted in computer science and computing but are applicable elsewhere; and 3) definitions bound to **educational and curricular frameworks** that essentially involve problem-solving approaches inspired by computer science or are applicable in computing.

Table 2 illustrates these three macro-categories by associating examples of single CT definitions drawn from the literature that are deemed to fall into the categorisation that Román-González and colleagues (2017a) propose. It should be noted that the examples provided under “Generic definitions” generally centre on thought processes. Those under “Educational and curricular definitions”, meanwhile, are drawn from sources bound to the educational sphere, and (hence) tend to place more emphasis on aspects like techniques, strategies, methods, knowledge building, understanding, efficiency – i.e., aspects with particular implications for the organisation and activation of learning processes. That said, they do share fundamental commonalities with generic definitions, given that they are indeed generic.

Categories of Computational Thinking definitions	Examples of Computational Thinking definitions In the analysed literature
Generic definitions	Computational thinking is the thought processes involved in formulating a problem and expressing its solution(s) so that a computer-human or machine can effectively carry it out (Grover & Pea, 2018)
	Computational thinking is about thinking processes, and its implementation is independent of technology (Hazzan et al., 2020)
	Computational thinking is defined as a thought process, *through skills that are fundamental in programming* (CT skills), to solve problems regardless of discipline (Zhang & Nouri, 2019)
Operational or model definitions	The framework of computational thinking involves solving problems, designing systems, and understanding human behaviour by drawing on the concepts fundamental to computer science (Jocius et al., 2020)
	Computational thinking is a set of broadly applicable problem-solving skills, including abstraction, decomposition, pattern recognition, and algorithmic thinking, among others (Huang & Looi, 2020)
	Computational thinking has eight core aspects: Abstraction, Algorithm Design, Evaluation, Generalization, Iterative Improvement, Information Representation, Precise Communication, and Problem Decomposition (Komm et al., 2020)
	Computational thinking definitions can be classified into four major categories: data practices, modelling & simulation practices, computational problem-solving practices, and systems thinking practices (Weintrop et al., 2016)
Educational and curricular definitions	Computational thinking involves systematically approaching problem-solving (e.g., algorithmically) in a manner that results in solutions that can be reusable in different contexts (Shute et al., 2017)
	Computational thinking is a problem-solving method that involves various *techniques and strategies* that can be implemented by digital systems (Australian Computing Academy, 2019)
	Computational thinking has been recognised for developing knowledge and understanding of concepts in Computer Science as well as for significant contribution to general-purpose problem-solving skills (Israel-Fishelson & Hershkovitz, 2020)
	Thinking computationally means being able to *approach and solve problems efficiently* based on the principles and methods of computer science (Arfé et al., 2020)

Table 2. Categories of Computational Thinking definitions from the literature
Source: Authors' elaboration

Researchers who propose definitions of Computational Thinking (CT) tend to regard it chiefly as a way of thinking that is intrinsically bound to problem solving (Hazzan et al., 2020; Zhang & Nouri, 2019) performed efficiently with the principles and methods of computer science (Arfé et al., 2020): "CT is NOT «thinking like a computer»; rather, it is about thinking like a computer scientist. It's the problem-solving approaches commonly used by computer scientists that constitute computational thinking" (Grover & Pea, 2018, p. 22). Many researchers stress that CT-based solutions must be expressed in a way that allows processing by a computer or robot (Corradini et al., 2017).

Programming and Computational Thinking are deeply intertwined, and their dual association is well noted in the literature. Programming supports the development of CT, while CT provides programming with a new upgraded role (Metcalf et al., 2021; Tikva & Tambouris, 2021). The distinction between the two is subtle in principle: CT does not necessarily require programming, although in practice, representing a solution to a problem as a program provides a perfect way to evaluate that solution. The computer will execute the instructions and, in doing so, provide the student with opportunities to refine their solution so that it is very precise (Webb et al., 2017). So CT is not necessarily about programming; instead, the emphasis is on (often computationally inspired) problem solving that promotes learning experiences (Hazzan et al., 2020).

Much CT-oriented research revolves around learning programming to acquire CT concepts and skills, but Computational Thinking also encompasses thought processes essential for problem solving in disciplines beyond computer science. The respective emphasis on these two sides of the coin is often context-dependent. For example, Taslibeyaz et al. (2020) reveal that in studies focusing on computer use, CT definitions concentrate on programming skills, whereas elsewhere thinking skills are given more prominence (see Table 3). Specific thought processes commonly linked with Computational Thinking include: i) creative problem solving; ii) algorithmic approach to problem solving; iii) problem-solution transfer; iv) logical reasoning; v) abstraction; vi) generalisation; vii) representation and organisation of data; viii) systemic thinking; ix) evaluation; and x) the social impact of computation (Fessakis & Prantsoudi, 2019; Román-González et al., 2017a; Sáez-López et al., 2016; Upadhyaya et al., 2020).

Computation Thinking associated with generic problem solving	Computation Thinking associated with programming and computing
Abstraction	Algorithmic Thinking
Data Analysis	Algorithm Design
Data Collection	Automation
Data Representation	Boolean Logic
Decomposition	Computation
Efficiency	Computational Modelling
Evaluation	Conditionals
Generalisation	Data Types
Logics & Logical Thinking	Events
Modelling	Functions
Patterns & Pattern Recognition	Iteration
Repeating Patterns	Loops (Repetition)
Simulation	Modularisation
System Thinking	Parallelisation
Visualisation	Sequencing
	Testing & Debugging
	Threads (Parallel Execution)

Table 3. Concepts concerning Computational Thinking skills development, as derived from the study's literature review and case studies
Source: Authors' elaboration

Despite the wide variety of definitions in use, it is possible to identify a set of Computational Thinking core concepts: abstraction, algorithmic thinking, automation, decomposition and generalisation (Curzon et al., 2019). These concepts are related to a set of attitudes and skills (or practices), including creating computational artefacts, testing and debugging, collaboration and creativity, and the ability to deal with open-ended problems (Grover & Pea, 2018). This understanding frames CT as a foundational competence for an informed citizen capable of coping with societal challenges. CT also bears potential as a means for creative problem solving and innovative approaches in several disciplines. It therefore has a crucial role to play in compulsory education. Programming/coding provides a laboratory for teaching and learning Computational Thinking – it makes CT concepts concrete. So Computational Thinking can become a tool for learning, e.g., as a medium for exploring different domains or self-expression (Resnick, 2017). This confirms the results of the 2016 EU Computational Thinking study, where CT is characterised by a set of core skills (Bocconi et al., 2016).

3.2 Relationship with Computer Science, Informatics and Computing

Researchers and practitioners stress the need to consider the practical implications of Computational Thinking, particularly its role in education (Curzon et al., 2019). Indeed, Fessakis and Prountsoundi (2019) argue that the term Computational Thinking was proposed as a conceptual vehicle to facilitate dialogue on the role of Computer Science/Informatics[8] in general education. As Jocius et al. (2020) state, "the value of computational thinking is not just as an isolated concept that relates to computer science, but also as a way to enhance and support more complex discipline-specific and interdisciplinary understandings". (p. 926). According to Kale et al. (2018, p. 575), teaching CT should "entail the knowledge of using computational thinking tools (technology), knowing which instructional strategies to use to teach computational thinking and the subject matter (pedagogy), and understanding of computational thinking and the subject matter (content)." Hsu et al. (2019, p. 261) draw on the results from their wide-ranging international review of CT policy-making in education to conclude that the different frameworks for understanding Computational Thinking reflect the variety of ways CT education policies have been envisioned globally.

Resonance can be found between the Computational Thinking definitions from the research sphere, as reported above, and a number of the curricula-based CT definitions proposed by respondents to the survey of Ministry of Education representatives that was conducted for this study (see Section 4.2). In Ireland, for example, CT is considered "a thought process (or human thinking skill) that uses analytic and algorithmic approaches to formulate, analyse and solve problems". At the same time, in Serbia, it is seen as a "thought process that involves formulating problems and their solutions so that the solutions are represented in a form that an information-processing agent can effectively carry out". In Singapore, a pragmatic view of CT is taken: "a thought process that involves formal reasoning, logical and algorithmic thinking, and the reformulation of a problem (for) a computer-based solution". Similarly, in Slovenia CT is seen as "thought processes involved in defining a problem and expressing its solution in a way that the solution can be effectively implemented by a computer". Significantly, however, the Slovenian definition goes on to embrace a broader, transversal dimension: "... [Computational Thinking is] transferable to other professional and scientific fields, contributes to the development of metacognitive skills and better problem solving in general".

Different understandings of Computational Thinking are also expressed by the participants interviewed for the in-depth Case Studies (see Annex 4). For example, the definitions of CT from Poland and Croatia both put a particular focus on problem solving. But in Poland, CT tends to be used in policy documents and among experts, while teachers and students more commonly use the term problem solving. In the UK-England, the term Computational Thinking is more common at primary level, while programming is more widely used at the secondary level. The main understanding of CT at ISCED 2[9] level revolves around three focus points: programming, algorithm, and problem solving (see discussion in Section 5.2).

8. We acknowledge the different connotations and variety of definitions of Computer Science and Informatics. However, in the context of this study, the two terms are used interchangeably, as in compulsory education they involve a common set of basic concepts (Box1).
9. International Standard Classification of Education: http://uis.unesco.org/en/topic/international-standard-classification-education-isced

The case study experts stressed that the term Computational Thinking does not reflect the proficient subject knowledge involved; they also emphasised that CT is not unpinned by a foundational subject area. Moreover, including the development of CT skills in curricula entails dealing with Computer Science/Informatics concepts on the one hand and addressing digital competence/digital literacy on the other. Investigation of evolution in Computational Thinking conceptualisation reveals that:

- discussions on Computational Thinking have motivated and given direction to the integration of computing in compulsory education;
- Computational Thinking skills provide tools for understanding and being an active member of our technology-infused social world;
- Computational Thinking education comprises mental skills and practices for
    a. designing computations that get computers to do jobs for us, and
    b. explaining and interpreting the world as a complex of information processes (infosphere).

The relationship between Computational Thinking and Computer Science/ Informatics in compulsory education is grounded in the fact that CT skills develop by learning basic Computer Science/Informatics concepts related to algorithms and programming.

4

Major trends in CT integration with compulsory education

Computational Thinking (CT) skills are being integrated into curricula across Europe and beyond. This section provides an overview of the current situation in **29 European countries**[10]. These include 27 countries[11] for which data were collected via the study's policy-level survey (see Section 2.2) and two countries, (UK) England and Sweden, for which data were collected via in-depth case studies. Results of a similar survey were also published in the previous 2016 CompuThink report (Bocconi et al., 2016). Where data from the two sets of results are comparable (common survey question, common responding countries), the evolution from 2016 to 2021 will be reported in this section. Indeed, representatives from 18 countries[12] replied to the 2016 CompuThink report survey, with 15 countries[13] answering both the 2016 and 2021 surveys.

Overall, **25 countries**[14] out of the 29 addressed in the study have already **included CT skills as part of their current statutory compulsory education curricula**, which in some cases has been approved and in others is already in place. This process has gained strong momentum in recent years, **with 18 of the 25 countries renewing their curricula between 2016 and 2021** (see the orange and light blue countries shown in Figure 4). The remaining countries are either (a) running school pilots – in DK – or finalising a draft curriculum – in BE fr – (shown in Figure 4 in dark yellow) or (b) have a predefined strategic plan to integrate CT skills (CZ, IT, SI – shown in green).

Note: *Curriculum* is an overloaded term that can refer to a *policy-related curriculum* and a *conceptual-pedagogical curriculum*. This study focuses on policy initiatives which have been officially accepted and that entail reform to national curricula and/or to official guidelines where the integration of CT skills in compulsory education has already been approved or enacted.

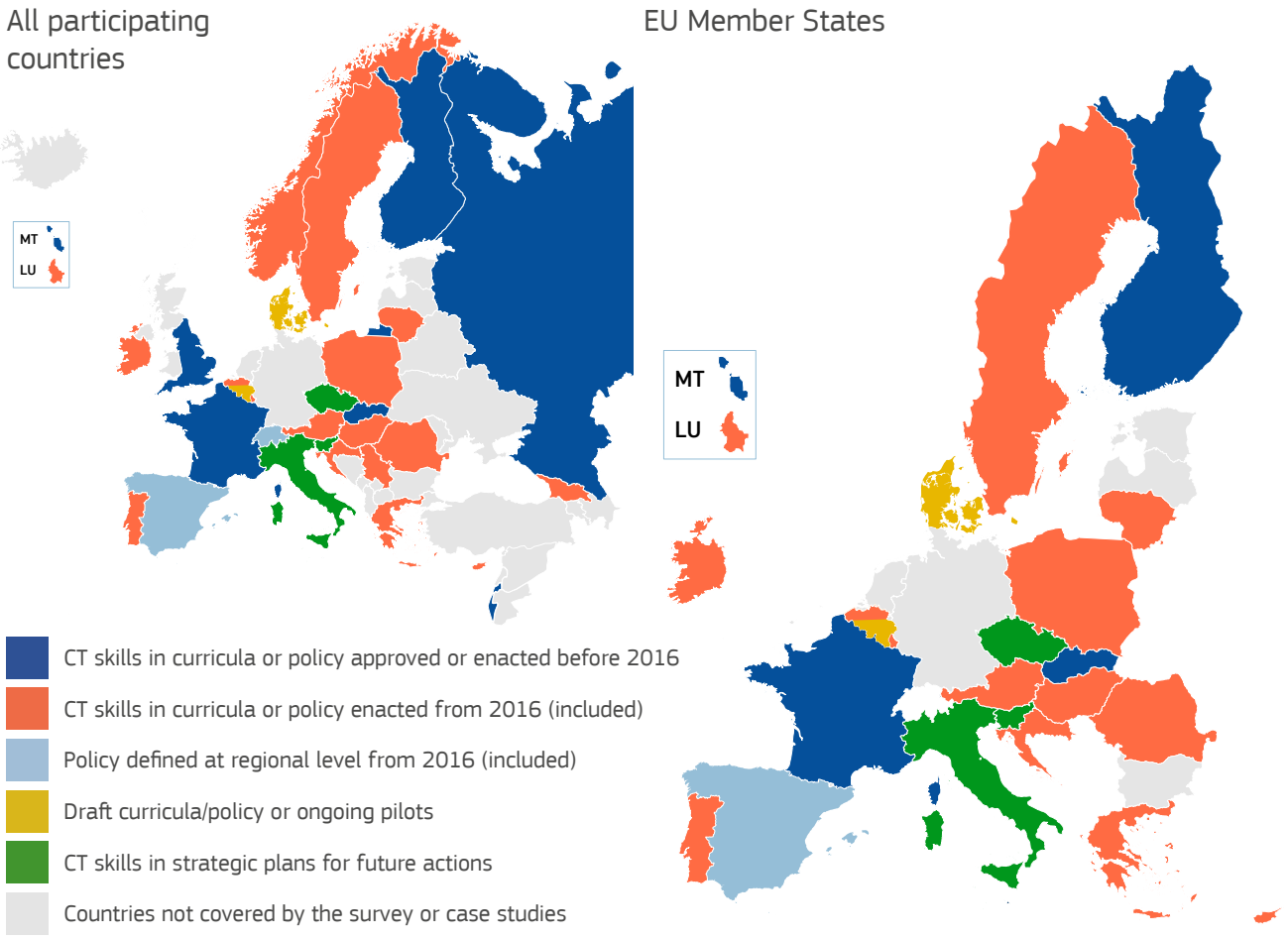


Figure 4. Overview of the state of CT skills integration in the compulsory education curricula of the 29 analysed countries
Source: Authors' elaboration based on results from the survey, desk research, and in-depth case studies

10. The study's policy-level survey also involved a non-European country, namely Singapore, whose data are reported on page 50.
11. Represented by 28 different respondents, with both Belgium Flanders (BE nl) and Belgium Wallonia (BE fr) replying to the survey.
12. Austria, Czech Republic, Denmark, Estonia, Finland, France, Hungary, Italy, Lithuania, Malta, Poland, Portugal, Spain, Sweden - and Israel, Norway, Switzerland, Turkey.
13. Austria, Czech Republic, Denmark, Finland, France, Hungary, Italy, Lithuania, Malta, Poland, Portugal, Spain - and Israel, Norway, Switzerland.
14. The 25 countries comprise 18 EU Member States (AT, BE nl, CY, EL, ES, FI, FR, HR, HU, IE, LT, LU, MT, PL, PT, RO, SE, SK) and 7 other European countries (CH, GE, IL, NO, RS, RU, UK-ENG). In ES and CH curricula are defined at regional level.

4.1 The rationale for including CT skills in curricula and official guidelines

Countries generally have multiple reasons for integrating CT skills into their curricula. Fostering problem-solving and logical-thinking skills emerge as reasons put forward by most countries. Almost all surveyed countries[15] aim to foster problem-solving skills, promote programming and coding,[16] and develop logical thinking skills.[17] These three predominant rationales reflect an understanding of CT as foundational for general education and so are in keeping with the endeavour to promote all students' CT skills. The second major cluster of reasons for including CT skills in compulsory education regards fostering employability. Eighteen out of the 29 countries considered[18] (almost two out of three) declare that they aim to support employability in this way, and 15 countries[19] aim to attract students to the study of Computer Science.

Country	Attracting more students to studying computer sciences	Fostering employability in the digital sector	Fostering coding and programming skills	Fostering problem-solving skills	Fostering logical thinking skills	Fostering other key competences	Other
Austria	✓	✓		✓			
Belgium Flanders			✓			✓	✓
Belgium Wallonia			✓	✓	✓		
Croatia			✓	✓	✓	✓	✓
Cyprus	✓	✓	✓	✓	✓		
Denmark			✓	✓	✓	✓	✓
Finland		✓	✓	✓	✓	✓	
France			✓	✓	✓	✓	
Greece	✓	✓	✓	✓	✓	✓	
Hungary	✓	✓	✓	✓	✓		
Ireland	✓	✓	✓	✓		✓	
Italy				✓	✓	✓	
Lithuania	✓	✓	✓	✓	✓		✓
Luxembourg	✓	✓	✓	✓	✓		✓
Malta		✓		✓	✓		
Poland	✓	✓	✓	✓	✓	✓	
Portugal			✓	✓	✓	✓	
Romania	✓	✓	✓	✓	✓	✓	
Slovakia	✓	✓	✓	✓	✓	✓	✓
Slovenia	✓	✓	✓	✓	✓	✓	
Spain	✓	✓	✓	✓	✓	✓	
Georgia			✓	✓	✓		
Israel	✓	✓	✓	✓	✓	✓	✓
Norway		✓	✓	✓	✓	✓	
Russia	✓	✓	✓	✓	✓		
Serbia			✓	✓	✓	✓	
Switzerland	✓	✓	✓	✓	✓	✓	

15. Except for Belgium Flanders
16. Except AT, IT, MT
17. Except for AT, BE nl, IE
18. AT, CY, EL, ES, FI, HU, IE, LT, LU, MT, PL, RO, SI, SK – and CH, IL, NO, RU.
19. AT, CY, EL, ES, HU, IE, LT, LU, PL, RO, SI, SK – and CH, IL, RU.

Table 4. Rationale for integrating Computational Thinking skills in the curriculum
Source: Authors' elaboration based on the survey results

By integrating CT skills in the compulsory curriculum, many countries aim to foster several transversal skills like critical thinking (EL, HR, LT, LU, RO, SK), creativity (EL, FR, LT, LU, PT), communication (LT, PT, SK), collaboration (EL, FR, PT, SK), personal development (HR, SK, RO), and analytical skills (EL, IT, RO). For example, in Finland the goal is to foster the seven key transversal competences in the Finnish curriculum.[20] In Ireland, CT activities can also foster design and development skills, digital media literacy and awareness, and numeracy skills. In Switzerland, one goal of CT integration is to develop students' analytical thinking and creative problem-solving.

Other goals mentioned in this light are fostering general digital skills (BE nl, RO, SI – and NO, RS). Several countries also mention more specific digital skills like the ethical use of new technologies (DK, FR, PL, SK). In Poland, students are expected to be provided with "a set of skills to interact with new technologies" and in Romania to "use technology interactively", while in Serbia there is a focus on "digital literacy, digital and technology-based competences, and digital safety". In Denmark, the curriculum also refers to areas like privacy, security, ethics, and interaction design. A few countries also give other reasons for integrating CT in their curricula. For instance, some countries see CT activities as a way to develop interest and/or competences in mathematics (LU, RO, SI), science education (BE nl, LU), and technology & engineering (LT, LU, RO, SI).

In conclusion, **the main rationale for introducing CT in most countries is to foster 21st century skills**, which are understood as essential for an active life in the digital world. This also emerged as the main motivation in the 2016 CompuThink report (Bocconi et al., 2016).

4.2 Positioning in the curriculum

This section investigates the position of CT skills in curricula, considered in terms of two criteria: education level (i.e., primary, lower secondary, upper secondary, initial VET); and subject(s) involved. In addition, it examines whether CT skills are integrated as a part of compulsory or elective subjects in the curriculum. Initial VET education is explored separately in Section 4.6.

As shown in Table 5, curricula in almost all the surveyed European countries (27 out of 28) refer to programming/coding. Most refer to Algorithmic Thinking (22 curricula), followed by Computational Thinking (16 curricula), Computer Science education (11 curricula) and Computing education (5 curricula). Eight curricula (AT, CY, HR, IE, LT, RO, SI – and CH) position Computational Thinking as part of Informatics and Computer Science subjects. Four curricula (AT, ES, RO, SI) also mention specifically that Computational Thinking is seen as a part of Computer Science and Informatics, but not exclusively limited to that, as Computational Thinking can also be part of other subjects (mainly STEM subjects).

Twenty-one curricula (BE fr, BE nl, HR, CY, DK, FI, FR, EL, HU, IE, LT, LU, PL, PT, RO, SK, SL – and GE, NO, RS, RU) refer both to programming and Algorithmic Thinking, indicating that they see CT skills development as being set within a context identified by the combination of algorithms and programming. Thus, one major trend is that basic concepts of Computer Science (including algorithms and programming) pave the way for developing CT skills.

20. Thinking and learning to learn; cultural competence, interaction and self-expression; taking care of oneself and managing daily life; multi-literacy; ICT competence; working life competence and entrepreneurship; participation, involvement and building a sustainable future.

Country	Programming / Coding	Algorithmic Thinking	Computational Thinking	Computer Science Education	Informatics Education	Computing Education	Other
Austria	✓		✓		✓		
Belgium Flanders	✓	✓	✓				
Belgium Wallonia	✓	✓					
Croatia	✓	✓	✓		✓		
Cyprus	✓	✓	✓				
Czech Republic	✓		✓	✓	✓		
Denmark	✓	✓	✓	✓	✓		
Finland	✓	✓	✓				
France	✓	✓	✓	✓			
Greece	✓	✓	✓	✓	✓	✓	
Hungary	✓	✓					
Ireland	✓	✓	✓	✓			
Italy	✓		✓		✓		
Lithuania	✓	✓	✓		✓		✓
Luxembourg	✓	✓	✓	✓			
Malta	✓					✓	
Poland	✓	✓	✓		✓	✓	
Portugal	✓	✓	✓				
Romania	✓	✓		✓	✓	✓	
Slovakia	✓	✓	✓	✓	✓	✓	
Slovenia	✓	✓		✓			
Spain	✓						✓
Georgia	✓	✓		✓			
Israel							✓
Norway	✓	✓					
Russia	✓	✓			✓		
Serbia	✓	✓		✓	✓		
Switzerland	✓	✓			✓		

Table 5. Relevant terms used in compulsory education curricula
Source: Authors' elaboration based on results from the study's survey and desk research

4.2.1 Curriculum location and integration of CT per education level

Three main approaches are followed when it comes to integrating CT skills in compulsory education curricula (Bocconi et al., 2016):

- as a cross-curricular theme – basic Computer Science (CS) concepts are addressed in all subjects, and all teachers share responsibility for developing CT skills;
- as part of a separate subject – basic CS concepts are taught in a computing-related subject (e.g., Informatics);
- within other subjects – basic CS concepts are integrated within other curriculum subjects (e.g., Maths and Technology).

In most countries, a combination of these approaches is in place. For example, in several countries CT skills are fundamentally developed as part of a Computer Science/Informatics subject and by embedding and extending those CT-related concepts across other subjects. In this light, this section provides information on, and an overview of, those 25 countries that have already introduced CT skills in their policy-related curricula in compulsory education,[21] i.e., primary (ISCED1) and lower secondary (ISCED2) – see countries shown in dark and light green in Figure 4 at the beginning of Section 4. Moreover, the section also looks at countries introducing CT skills in upper secondary (Section 4.5) and initial VET (Section 4.6).

Out of the 25 countries that have already included CT skills in their curricula, 22 have done so at the primary education level. In 19 of these 22 countries, CT skills are developed as part of compulsory subjects (AT, CY, EL, FI, FR, HU, LT, LU, MT, PL, PT, SE, SK – and CH, GE, NO, RS, RU, UK-ENG), while in three cases (ES, HR – and IL)[22] they are part of elective subjects. As shown below in Figure 5, ten countries (depicted in blue or with a blue-backed pattern) have integrated CT skills primarily as part of a *separate subject* (CY, EL, HR, HU, LT, PL, SK – and IL, RS, UK-ENG), five countries *within other subjects*, e.g., Maths, Technology (FI, FR, SE – and GE, NO) and six countries as a *cross-curricular theme* (AT, EL, LU, MT, PT – and RU). In Spain and Switzerland, autonomous regions have integrated CT skills either within other subjects (e.g., Maths) or as a cross-curricular theme. In several of these countries, approaches have been combined: in five countries (FI, LU, PT, SE – and RU), CT skills are developed as part of a *cross-curricular theme* and *within other subjects* (e.g., Maths, Technology); in another five countries (EL, HR, LT, PL, SK) they are *part of a separate subject* and also a *cross-curricular theme*; and in Cyprus, they are *part of a separate subject* and addressed *within other subjects*.

21. As mentioned above, this study focuses on policy initiatives that entail reform to national curricula (and/or to official education guidelines) and where integration of CT skills in compulsory education is already approved or being enacted. Therefore, five EU countries that replied to the study (namely Belgium Wallonia, Czech Republic, Denmark, Italy, and Slovenia) are not discussed here in Section 4.2.1, as they are either yet to make an official policy decision or have not yet approved a curriculum in which CT skills are integrated into compulsory education.
22. The issue of whether to integrate CT skills as part of a compulsory or elective element in the curriculum is closely related to the rationale for integrating CT (e.g., ensuring all students have the skills needed for life in a digital world, attracting gifted students to pursue computer-related careers), and to the question of equity (e.g., the proportion of boys and girls acquiring CT skills).

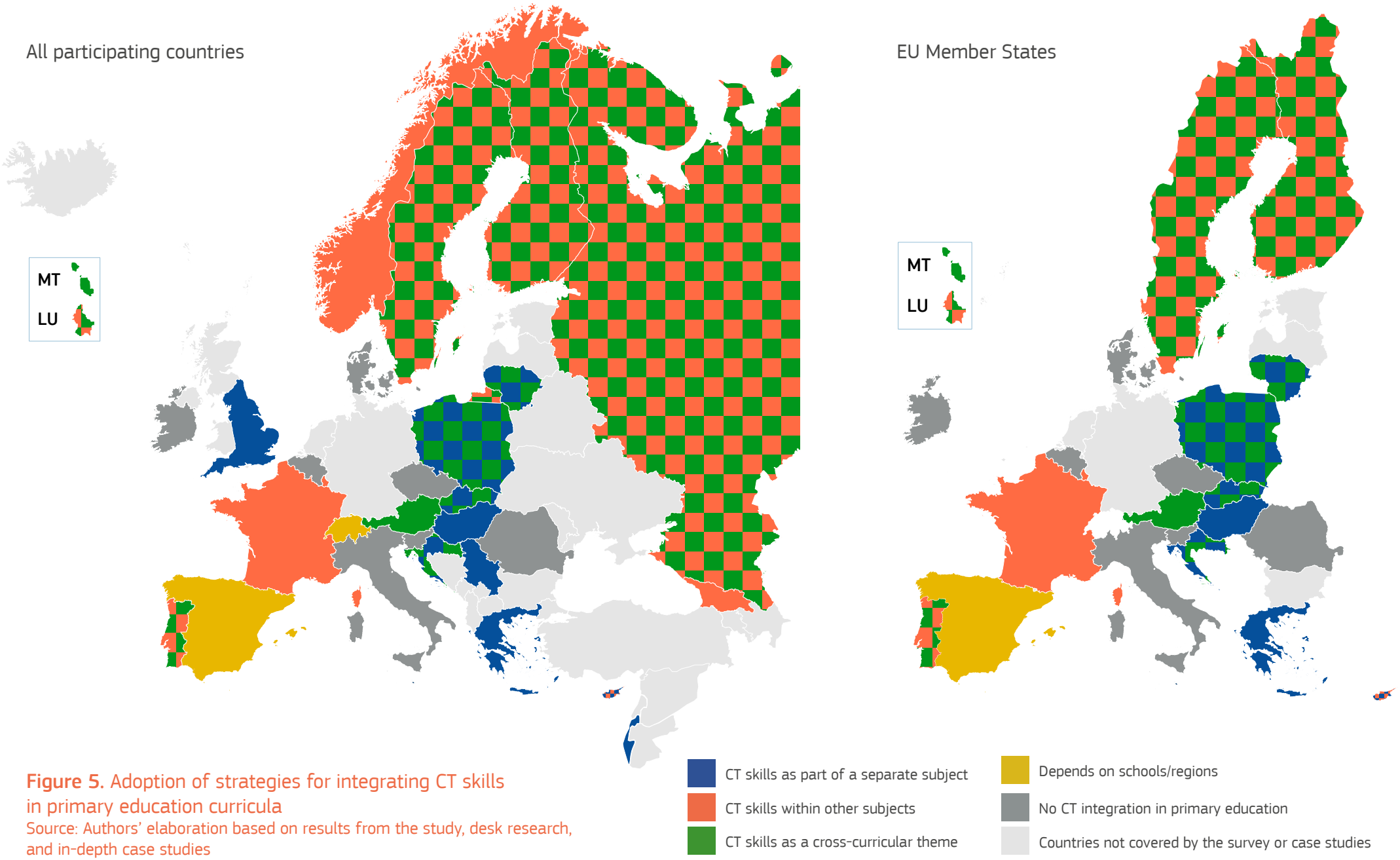


Figure 5. Adoption of strategies for integrating CT skills in primary education curricula
Source: Authors' elaboration based on results from the study, desk research, and in-depth case studies

At the lower secondary school level, **24 countries have already included the development of CT skills in their curricula**. Of these, 20 have done so as part of compulsory subjects (AT, BE nl, EL, HR, HU, FI, FR, LT, LU, MT, PL, PT, RO, SE, SK – and CH, NO, RS, RU, UK-ENG) and four as elective subjects (HR, IE – and GE, IL). In 16 countries (shown in blue in Figure 6), CT skills are primarily integrated as *part of a separate subject* (AT, EL, HR, HU, IE, LT, LU, MT, PL, RO, SK – and CH, IL, RS, RU, UK-ENG). In six countries, they are addressed *within other subjects*, e.g., Maths, Technology (FI, FR, PT, SE – and GE, NO), and in five countries as a *cross-curricular theme* (BE nl, EL, FI, PT, SE). In some of these countries, approaches are combined: in Finland, Portugal and Sweden, CT skills are developed as part of a *cross-curricular theme* and *within other subjects* (e.g., Maths, Tech). This combination is depicted in Figure 6 by the orange-green or orange-blue pattern. In Austria, Belgium Flanders, Luxembourg, Spain and Switzerland, curriculum location depends on curricula defined at the regional or school level.

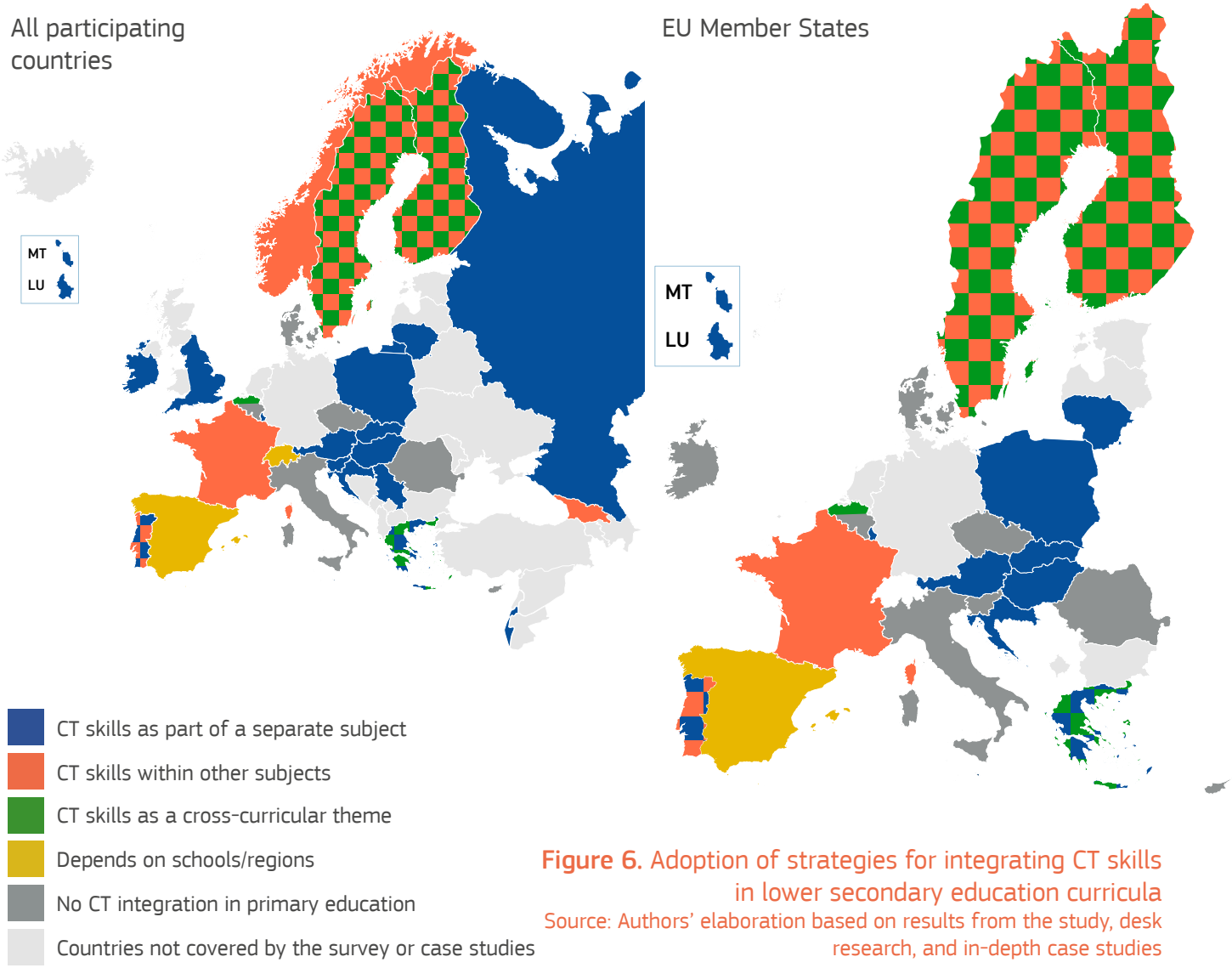


Figure 6. Adoption of strategies for integrating CT skills in lower secondary education curricula

Source: Authors' elaboration based on results from the study, desk research, and in-depth case studies

In most of the surveyed countries, the compulsory education curricula, both at primary and lower secondary levels, pursue the **development of CT skills through basic Computer Science (CS) contents, referring to a combination of *algorithms* and *programming***. Moreover, as discussed in the previous sections, CT skills are invariably coupled with digital competence/digital literacy elements. This is a major trend in all the surveyed compulsory education curricula. For example, when basic CS concepts are positioned within a single subject, this same subject also encompasses contents related to digital competence/digital literacy. This is the case, for example, with the school subjects *Computing* in UK-England, *Computer Science* in Poland, *Basic Digital Education* in Austria, and *Digital Culture* in Hungary. In countries where basic CS concepts are positioned within existing subjects, the subject in question is predominantly mathematics, sometimes in conjunction with a subject like technology, as is the case with the French curriculum.

Eight countries responding to the survey (AT, CY, HR, IE, LT, SI, RO - and CH) position CT as part of Informatics and Computer Science in compulsory education. Three other countries (AT, ES, SI) express the same stance and extend CT to other subjects, especially STEM subjects. For instance, in **Croatia** CT is considered an essential part of the Informatics subject. The new Croatian curriculum states that the "focus of the educational process in the subject "Informatics" should be on problem solving and programming to help students develop Computational Thinking, which enables understanding, analysis and problem solving". In **Georgia**, Computational (Algorithmic) Thinking was introduced early on and is regarded as a foundational concept underpinning Computing / Computer Science / Informatics. In the view of the **Lithuanian** National Agency for Education, the core of the Informatics subject is Computer Science, in which students are taught the principles of information, data, algorithms and computation, how digital systems work, and how to put this knowledge to use through programming.

Some countries mention a relationship between CT and programming, although this aspect was not included in the survey conducted for this study. In **Finland**, programming is seen as a technical process/task carried out using a digital device and programming languages. Hence, programming is only part of Computational Thinking/Algorithmic Thinking. It can include dissembling a problem, recognising and processing patterns/formulas, programming, and automation. In **Serbia**, CT involves coding and algorithmic thinking. Coding entails using a computer language to solve a problem with a computer. Coding is usually taught during Informatics and Computer Science classes, and so this governs how the two terms are related.

In summary, among the 25 surveyed countries that have already introduced the development of CT skills in their statutory curricula (approved or enacted) for compulsory education, two main trends emerge:

1. CT is understood as a set of skills developed through basic CS concepts (algorithms & programming);
2. basic CS concepts (algorithms & programming) are coupled with digital competence and digital literacy elements.

4.3 Challenges posed by the integration of CT skills in compulsory education

Respondents to the study survey were asked whether any of four key challenges were faced at the different education levels and if other challenges arose. The four challenges presented were:

- **competition with other curriculum priorities;**
- **lack of adequately trained teachers;**
- **lack of tools and resources for teaching;**
- **difficulties in assessing Computational Thinking/programming skills.**

Only education levels where CT skills development is actually integrated were considered, so the number of replies varies from level to level (see Table 6). Across education levels, most respondents mentioned the *lack of adequately trained teachers* as a challenge (18 countries at primary and 21 at lower secondary level). By comparison, *competition with other curriculum priorities* was identified as slightly more of a challenge at primary (13) than at the lower secondary level (11). Regarding primary education, these findings are in line with results from the literature review and the case studies, which found that teachers are not adequately prepared to integrate the development of CT skills in their teaching, and experience difficulties in adding new elements to an already crowded curriculum. *Assessment of Computational Thinking / programming* skills emerged as a slightly stronger challenge at lower secondary (13) than at primary level (10). One possible interpretation of the difficulties experienced in assessing CT skills is that several countries have final national exams at the end of the lower secondary cycle, and skills like CT can be difficult to quantify in such high-stakes assessments. Finally, the *lack of tools and resources for teaching* (nine countries in primary and seven countries in lower secondary schools) represents less of a challenge for countries than the other issues.

Legend

- Competition with other curriculum priorities
- Lack of adequately trained teachers
- Lack of tools and resources for teaching
- Assessment of CT skills
- N/A — No integration at this level

Country[23]	Primary school level			
Austria	–	✕		
Belgium Flanders	N/A	N/A	N/A	N/A
Croatia		✕		
Cyprus	–	✕	⚙	✓
Denmark	–	✕		✓
Finland	–	✕	⚙	✓
France	–	✕		
Greece	–	✕	⚙	✓
Hungary		✕		
Ireland	N/A	N/A	N/A	N/A
Lithuania	–	✕	⚙	✓
Luxembourg	–	✕	⚙	
Malta				
Poland		✕	⚙	✓
Portugal		✕		
Romania	N/A	N/A	N/A	N/A
Slovakia		✕	⚙	✓
Spain	–			✓
Georgia		✕		
Israel	–			
Norway	–	✕		✓
Russia		✕	⚙	
Serbia	–	✕	⚙	✓
Switzerland	–	✕		

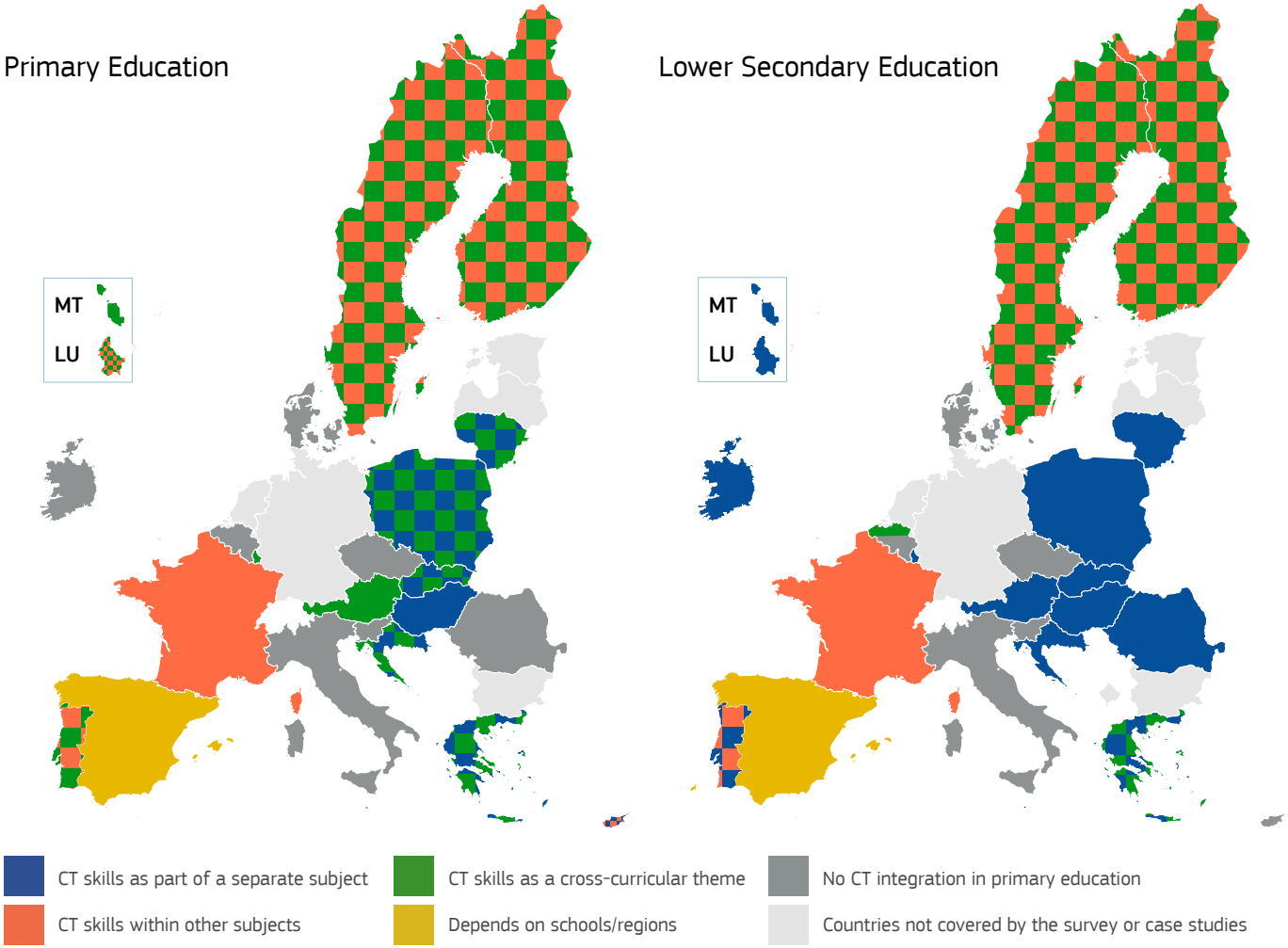
Country	Lower secondary level			
Austria		✕		✓
Belgium Flanders		✕		✓
Croatia		✕		
Cyprus	N/A	N/A	N/A	N/A
Denmark	–	✕		✓
Finland	–	✕	⚙	✓
France	–	✕		
Greece		✕	⚙	✓
Hungary		✕		
Ireland	–	✕		
Lithuania	–		⚙	✓
Luxembourg		✕		
Malta	–	✕		✓
Poland		✕	⚙	✓
Portugal		✕		
Romania	–	✕	⚙	✓
Slovakia		✕	⚙	✓
Spain	–	✕		✓
Georgia		✕		
Israel	N/A	N/A	N/A	N/A
Norway	–	✕		✓
Russia		✕		
Serbia	–	✕	⚙	✓
Switzerland	–	✕		

Table 6. Challenges related to the integration of Computational Thinking skills development in compulsory education
Source: Authors' elaboration based on results from the study survey

23. This table presents data from the survey responses received from 23 countries that have already included CT skills as part of their (approved or enacted) statutory curricula. In addition, data from Denmark (where a pilot is in place) have also been included in the table, making a total of 24 countries.

4.4 CT skills in the compulsory education curricula of different countries

This section provides a brief look at how the development of Computational Thinking skills in compulsory education is treated in the analysed countries. Figure 7 shows integration of CT skills in EU Member States' primary and lower secondary curricula.



EU Member States participating in the study	AT, BE, CY, CZ, DK, EL, ES, FI, FR, HR, HU, IE, IT, LT, LU, MT, PL, PT, RO, SE SI, SK	22
CT skills integrated in ISCED 1	AT, CY, EL, ES, FI, FR, HR, HU, LT, LU, MT, PL, PT, SE, SK	15
CT skills as a cross-curricular theme in ISCED 1	AT, MT, EL, HR, LT, PL, SK, FI, LU, PT, SE	11
CT skills as a cross-curricular theme in ISCED 2	BE nl, EL, FI, PT, SE	5
Renewed curricula between 2016 & 2021 including CT skills	AT, BE nl, CY, EL ES, HR, HU, IE, LT, LU, PL, PT, RO, SE, SK	15
CT skills integrated in ISCED 2	AT, BE nl, EL, ES, HR, FL, GR, HU, IE, LT, LU, MT, PL, PT, RO, SE, SK	17
CT skills as part of a separate subject in ISCED 1	CY, EL, HR, HU, LT, PL, SK	7
CT skills as part of a separate subject in ISCED 2	AT, EL, HR, HU, IE, LT, LU, MT, PL, RO, SK	11
CT skills as a compulsory element in both ISCED 1 & 2 curricula	AT, EL, FI, FR, HU, LT, LU, MT, PL, PT, SE, SK	12
CT skills integrated in both ISCED 1 & 2	AT, EL, ES, FI, FR, HR, HU, LT, LU, MT, PL, PT, SE, SK	14
CT skills within other subjects in ISCED 1	CY, FI, FR, LU, PT, SE	6
CT skills within other subjects in ISCED 2	FI, FR, PT, SE	4

Figure 7. Overview of CT skills integration in EU Member States' primary and lower secondary education system
Source: Authors' elaboration based on results from the survey, desk research, and in-depth case studies

The country-by-country overview begins with a glimpse at the situation in 22 European Union Member States, then moves on to seven other European countries (Georgia, Israel, Norway, Russia, Serbia, Switzerland, UK-England), and finally considers a country outside the European sphere, namely Singapore.

Austria

In Austria, CT skills development has been part of the curriculum since 2016. The curriculum refers to CT, to programming but also to Informatics Thinking. CT is integrated as a compulsory element across all levels of general education. In primary (since 2021) and lower secondary school (since 2018), CT has been part of the subject "Basic Digital Education". This covers the topic "Computational Thinking" (i.e., working with algorithms and creative use of programming languages), along with information technology and media-related competences. Regional education and school authorities have the prerogative to autonomously introduce CT skills into any subject, depending on the educational policies and strategies they opt to implement. From grade 9 (upper secondary), the subject Informatics is mandatory. The current subject "Digitale Grundbildung" is put into practice by schools autonomously. As of the 2022/23 school year, this subject will be replaced by the new compulsory subject "Digitale Grundbildung", which will be taught the same way in one lesson per week in grades 5-8 in all schools following a unified curriculum.[24] An evaluation of the current curriculum is underway but has yet to be completed at the time of writing.

Belgium Flanders and Belgium Wallonia

The terms Computational Thinking, Algorithmic Thinking and programming/coding are used in Belgium Flanders curricula. CT is defined as the human capacity (power) to solve complex problems with the help of the computer. Some education providers also issue specifications on how to teach CT in their specific curriculum (see for example the Catholic Education provider[25]). Since 2019, transversal Digital Competence (including "CT & practices") has been included at the lower secondary level. CT skills are also integrated into the curriculum to raise interest in science education. CT skills are also integrated into maths, geography (processing of GIS data), sciences, and STEM taught in upper secondary schools. The actual integration of CT skills in individual school curricula is a matter for schools to decide.

In the French Community of Belgium, a new common core curriculum is to be approved by the government. In this new curriculum, CT skills will be addressed within a separate subject that will be compulsory in primary and lower secondary schools. The document "Le Pacte pour un Enseignement d'excellence" provides the framework for this reform. The goal is to foster students' coding and programming skills and their logical and problem-solving skills. The new curriculum explicitly refers to the terms Algorithmic Thinking and programming/coding.

Croatia

In the new 2018 curriculum,[26] the subject Informatics – and therefore CT skills – has become part of all twelve years of general education. Since the 2020-2021 school year, primary schools have also had the option to offer Informatics in grades 1, 2, 3, 4, 7, and 8. The subject Informatics is compulsory in Croatia in grades 5 and 6. Previously, it was an elective subject in grades 5 to 8. As well as integration in Informatics, CT skills are also part of other subjects, e.g., Mathematics, Croatian language, Nature & Society in primary education, and Arts. The current Informatics curriculum builds on Croatia's long tradition in Computer Science. It covers four domains: information and digital technologies; CT and programming; digital literacy and communication; and e-society.

24. Sommerschule: Freiwilliger Förderunterricht wird zur jährlichen Regel | Pressedienst der Parlamentsdirektion – Parlamentskorrespondenz, 15.12.2021 (ots.at)
25. https://zill.katholiekonderwijs.vlaanderen/#!/leerinhoud/WD/lw/7
26. Curriculum reform introduced in 2018 made it compulsory for schools to provide two hours of Informatics per week in Grades 5 and 6 and introduced it as an elective subject in Grades 7 and 8. Since the 2020/21 school year, primary schools have also had the option to offer Informatics in Grades 1 to 4.

Cyprus

In Cyprus, the term "Ypologistiki Skepsi" refers to Computational Thinking, but there is no officially sanctioned definition of the term. However, certain level indicators have recently been defined for the curriculum of the subject "Design Technology – Digital Technology Education"[27] for students aged nine to eleven. Since 2019, CT skills have been taught at the primary level as part of this subject and as part of Mathematics. In addition, in-service training is provided for Design Technology teachers to become able and confident in teaching the programming and robotics elements of the subject.

Czech Republic

CT skills are an integral part of the Strategy of Education Policy 2030+,[28] in which CT is explicitly mentioned under "Strategic line 1: Transforming the content, methods and assessment of education" (1.4 Digital Learning). According to this strategy document, students should be able to exploit the opportunities that digital environments offer and be aware of the potential risks related to technology use. The document also stresses that education should focus on developing students' unique skills that will not be replaced in the future by automation. As indicated in the strategy document, the review of framework curricula in the field of basic literacy is currently ongoing and expected to be completed by 2023 (Key Activity 1.3). The term "informatické myšlení" (Informatic Thinking) is also used in the Czech Republic.

Denmark

CT skills are part of a pilot program[29] started in 2019 called "Technology Comprehension", financed by Denmark's Ministry for Children and Education. This pilot aims to gather knowledge and experiences about whether and how technological comprehension can be taught in primary and lower secondary school. It is expected to start building capacity and competences in the educational sector so that teachers generally can work within this field and subject area. In the pilot, Technology Comprehension has been implemented as a subject in 46 primary and lower secondary schools. This combines aspects of Social Sciences, Computer Science and humanities. The subject is intended to develop the skills, insights and capacities children need to critically and constructively engage with digital technologies. Initial evaluation results have been published.[30] Once the pilot programme has concluded, a political decision will be taken on the subject's future in primary and lower secondary schools.

27. https://scheted.schools.ac.cy/index.php/el/schediasmos-technologia/analytiko-programma
28. Strategie vzdělávací politiky ČR do roku 2030+ - https://www.msmt.cz/vzdelavani/skolstvi-v-cr/strategie-2030
29. Faghæfte - Fælles Mål, læseplan og vejledning | emu danmarks læringsportal
30. https://xn--tekforsget-6cb.dk/om-forsoget/evaluering/

Finland

The national core curriculum[31] sets out how CT skills are integrated into education. The new literacy skills programme[32] identifies CT and programming jointly as one of the three main priority topics, the other two being media literacy and information and communication technologies (ICT). CT is described as a way to think and work with any kind of task. Integrating CT activities in the curriculum is intended to foster the seven key transversal competences in the Finnish curriculum.[33] In basic education (grades 1-9), CT skills are integrated across different subjects such as mathematics, crafts and environmental studies. The main responsibility for integrating CT skills lies on the shoulders of maths teachers. In upper secondary education, schools and municipalities can offer different sorts of courses related to programming, CT and CS.

France

CT skills were introduced in France in 2015 at the primary and lower secondary level, where CT concepts are taught as part of mathematics and technology subjects. In 2019, they were introduced at upper secondary level within the courses "SNT – Sciences numériques et technologie" and "NSI – Numérique et Sciences Informatiques". In France, the term "algorithmique et programmation" (Algorithmic Thinking and programming) is used in primary to upper secondary school curricula. Students are expected to develop Algorithmic Thinking and problem-solving skills and learn how to process and present information. Teachers are encouraged to use project-based, active, and collaborative pedagogical approaches. Since 2015, the "Diplôme national du brevet" (DNB) exam, which assesses knowledge and skills acquired by the end of lower secondary school, includes an exercise in informatics programming in mathematics, science and technology. Moreover, students' digital competence is assessed through the national online platform for the evaluation and certification of digital skills, called PIX.[34] This is based on DigComp 2.0 (Vuorikari et al., 2016).

Greece

As part of a major ongoing curricula reform, a compulsory subject called "ICT in education and informatics" has been integrated into the new primary education curriculum.[35] This subject (taught one hour per week from grades 1 to 6) encompasses Computational Thinking – problem solving, programming and digital competence. As part of this reform, Informatics is now taught two hours per week (instead of the previous one hour) as a separate compulsory subject at lower secondary education level. The main thematic areas covered are algorithms, programming, computer systems and networks, problem solving, data analysis, digital literacy and digital citizenship. Pilot implementation of the new curriculum was launched in the 2021-2022 school year, involving 112 schools. A larger-scale implementation will follow this in 2022, and extension to all schools will start from 2023-2024.

31. https://www.oph.fi/fi/koulutus-ja-tutkinnot/perusopetuksen-opetussuunnitelman-perusteet
32. Uudet lukutaidot, https://uudetlukutaidot.fi/osaamisen-kuvaukset/ohjelmointiosaaminen/
33. Thinking and learning to learn; Cultural competence, Interaction and self-expression; Taking care of oneself and managing daily life; Multiliteracy; ICT Competence; Working life competence and entrepreneurship; Participation, involvement and building a sustainable future.
34. https://orga.pix.fr/connexion; https://pix.fr/
35. http://iep.edu.gr/el/index.php?option=com_content&view=article&id=3388

Hungary

In Hungary, the term Algorithmic Thinking (*algoritmikus gondolkodás*) is used, which is usually understood as the ability to think in a sequence of steps (a process). The term is frequently used in relation to problem solving. The current national curriculum reform results from a general evaluation carried out at least every seven years. CT activities have been part of the curriculum in some form or other since the early 1990s, but in the most recently issued national curriculum (NAT 2020)[36] and related framework curriculum, the topic is addressed explicitly and systematically, with considerable recommended teaching time. The development of Algorithmic Thinking is a goal of the subject Digital Culture (under the mandatory Technology discipline). According to the 2020 framework curricula[37] for the subject, around a third of teaching time is devoted to CT-related topics like algorithms, robotics and programming. Digital Culture is taught from grades 3 to 11 as a mandatory subject. However, inclusion of this subject in the school leaving exam (*Matura*) is elective.

Ireland

In Ireland, CT skills have been part of the curriculum since 2016. They are also part of an elective Coding short course offered at lower secondary level. As part of the Computer Science specification for Senior Cycle students,[38] CT is defined as a "thought process (or a human thinking skill) that uses analytic and algorithmic approaches to formulate, analyse and solve problems". CT is a core process of Computer Science in the senior cycle and in the Coding short course[39] for Junior Cycle students. Guidelines on how to complete the coursework as part of the Computer Science subject are available.[40] A special service called PDST[41] provides educational support specifically for programming and CS. The current CS curriculum was rolled out in 2018, and a research group called LERO produced an analysis report[42] on Professional Development measures and their impact on teachers. The term CT is also used in the draft framework for the primary curriculum.[43] While this new curriculum has yet to be approved, the groundwork has been performed to build capacity in advance.[44]

Italy

CT skills are not part of Italy's official national curriculum guidelines for compulsory education. Nevertheless, the Ministry of Education has issued two major policy documents on digital education that address Computational Thinking: the National Strategy for Digital Schools[45] published in 2015 and the National Indications and New Scenarios[46] (for pre- and primary schools). In these documents, CT is treated as a key topic to be promoted in schools. In addition, the Ministry has supported initiatives both nationally (related to Code Week and code.org) and at the local level. Several primary schools are including CT skills, coding, and robotics activities in conceptual-pedagogical curricula they themselves have defined. At upper secondary level, Informatics is a curricular subject in schools that offer certain streams of secondary study.

36. https://magyarkozlony.hu/dokumentumok/3288b6548a740b9c8daf918a399a0bed1985db0f/letoltes
37. 2020 framework curricula for the subject Digital Culture (Digitális kultúra): Primary https://www.oktatas.hu/kozneveles/kerettantervek/2020_nat/kerettanterv_alt_isk_1_4_evf; Lower secondary https://www.oktatas.hu/kozneveles/kerettantervek/2020_nat/kerettanterv_alt_isk_5_8;
38. https://curriculumonline.ie/getmedia/d73af6e3-b4e5-4edb-a514-6383e2306a4b/16626-NCCASpecification-for-Leaving-Certificate-CS-WEB-v4.pdf
39. https://curriculumonline.ie/Junior-cycle/Short-Courses/Coding/
40. https://curriculumonline.ie/Senior-cycle/Senior-Cycle-Subjects/Computer-Science/
41. http://www.compsci.ie/
42. https://lero.ie/sites/default/files/LCCS%20PD%20Final%20Report%20August%202020.pdf
43. https://ncca.ie/media/4870/en-primary-curriculum-framework-dec-2020.pdf
44. https://ncca.ie/media/4155/primary-coding_final-report-on-the-coding-in-primary-schools-initiative.pdf
https://ncca.ie/media/3937/ncca-coding-in-primary-schools-initiative-research-paper-on-computational-thinking-final.pdf
45. https://www.miur.gov.it/documents/20182/50615/Piano+nazionale+scuola+digitale.pdf/5b1a7e34-b678-40c5-8d26-e7b646708d70?version=1.1&t=1496170125686
46. https://www.miur.gov.it/documents/20182/0/Indicazioni+nazionali+e+nuovi+scenari/

Lithuania

CT skills are currently part of the Information Technologies (IT) subject. In 2019, Lithuania's Ministry of Education, Science and Sport approved the Guidelines for Updating the General Curriculum Framework.[47] This foresees a new Informatics subject[48] to replace the current IT subject, with the introduction starting from primary school. A pilot implementation of the new curriculum involving a hundred primary schools commenced in 2019.[49] The new curriculum will come into effect in 2023.

Luxembourg

The curriculum in Luxembourg has included CT activities since 2020. These CT activities are compulsory elements at the primary level across all subjects. CT is described as focusing on developing problem-solving strategies such as algorithms. The emphasis is not so much on the learner solving the task at hand but on setting up a sequence of actions that could lead to a solution. A new subject called "Digital Sciences" has been introduced at the secondary level as part of a secondary school pilot scheme launched in September 2021. This new subject will become compulsory from September 2022 for all secondary schools. Support is provided via the websites kodeieren.lu and EduCoding.lu.[50] The former is a networking platform connecting teachers with experts and service providers, while the latter presents activities and lesson plans.

Malta

In Malta, the terms critical thinking and problem-solving skills are generally used. The Framework for the Education Strategy for Malta 2014-2024,[51] the Learning Outcomes Framework,[52] and the National Curriculum Framework[53] describe the curriculum, including activities to foster CT skills. The Learning Outcomes Framework (developed in 2015 and enacted in 2018) for Primary grades 1 to 6 covers the compulsory cross-theme Digital Literacy,[54] which includes the topic Computational Thinking. In lower secondary school, Computing[55] is taught as a compulsory subject in grades 8 and 9.

47. https://www.mokykla2030.lt/dokumentai/
48. https://www.mokykla2030.lt/informatikos-ugdymas/
49. https://informatika.ugdome.lt/lt/apie/
50. Home | SCRIPT Coding (kodeieren.lu), Kodéieren an der Grondschoul (educoding.lu)
51. https://education.gov.mt/en/resources/Documents/Policy%20Documents%202014/BOOKLET%20ESM%202014-2024%20ENG%2019-02.pdf
52. https://www.schoolslearningoutcomes.edu.mt/en/category/cross-curricular-themes
53. https://curriculum.gov.mt/en/Resources/The-NCF/Documents/NCF.pdf
54. Learning Outcomes Framework (schoolslearningoutcomes.edu.mt)
55. https://www.schoolslearningoutcomes.edu.mt/en/subjects/computing

Poland

Revision of the Polish core curriculum in 2017 led to the idea that Computer Science should start as a subject from grade 1 of primary school. Thus, Computer Science became compulsory from grades 1 to 12, whereas previously this was only the case in lower secondary education. The curriculum has been in place since 2017 in primary schools from grades 1 to 8. As of the 2019-2020 school year, the reform commenced in secondary schools, and curricular changes are still being implemented. The new curriculum is, in some parts, an extension of the previous one, seeking to unify aims at different levels, applying a more homogenous terminology, and repositioning activities under the Computer Science umbrella.

Portugal

Since 2018, CT skills have been an integral part of "ICT",[56] a compulsory topic integrated as a transversal theme at primary and lower secondary levels. In about 40 percent of school clusters, this mandatory element is complemented by optional robotics clubs for students interested in this field. In these clubs, students develop coding skills and undertake robotics projects on specific topics dealt with in their formal education, "Novas Aprendizagens Essenciais de Matemática",[57] a new curriculum for mathematics for grades 1-9. This new curriculum is to be enacted starting in 2022. It identifies six core transversal mathematical skills: problem solving, mathematical reasoning, mathematical communication, mathematical representations, mathematical connections, and Computational Thinking. The intention is to approach different dimensions of CT skills: problem solving, decomposition, debugging, pattern recognition, algorithms, programming, and robotics.

Romania

In 2017, Romania introduced a new curriculum[58] for lower secondary education that includes the subject Informatics, which is compulsory as of grade 5. While the term Computational Thinking is used in Romania, the curriculum does not refer to it directly. Nevertheless, it does mention relevant concepts such as abstraction, algorithmic thinking, automation, decomposition, and generalisation. The curriculum for upper secondary education is currently being revised. The Strategy for Digitalizing Education in Romania (2012-2027),[59] which is tied to the European Commission's Digital Education Action Plan (2021 – 2027), sets out general goals for digital education in Romania. This strategy was developed by the Ministry of Education and relevant bodies from industry and society at large. At the regional level, many Non-Governmental Organisations (NGOs) are involved in developing CT activities in schools.

56. https://www.dge.mec.pt/sites/default/files/Curriculo/Aprendizagens_Essenciais/1_ciclo/oc_1_tic_1.pdf
57. https://files.dre.pt/2s/2021/08/161000000/00115001116.pdf
58. http://programe.ise.ro/Portals/1/Curriculum/2017-progr/117-INFORMATICA%20si%20TIC.pdf
59. https://www.smart.edu.ro/

Slovakia

In Slovakia, CT activities are part of the compulsory Informatics subject.[60] This has been part of upper secondary education since 1984. The subject was integrated into lower secondary education in 2005 and primary education in 2008. The subject focuses in part on concepts generally associated with CT skills, such as algorithmic thinking, abstraction, generalisation, decomposition, and automation.

Slovenia

In Slovenia, the process of curricular revision started in 2021. It is co-financed with funds from the European Commission Recovery and Resilience Facility and will last at least two years. The term Computational Thinking (*Računalniško mišljenje*) is used in Slovenia and examples of CT-related activities are available from the "Computing and Informatics for all" website.[61] CT is one of the basic skill sets that students are expected to develop in Informatics lessons. CT skills can also be developed, consolidated and applied in other subjects. The Slovenian Digital Education Plan 2027, which also mentions CT, is currently under preparation. In July 2021, the Ministry of Education established a new division specifically dedicated to digital education, whose mission includes establishing collaboration with all stakeholders (e.g., NGOs, industry, experts) as part of a national digital coalition.

Spain

While Spain's national curriculum for primary education does not refer to Computational Thinking, it is currently under review, and it is expected that CT skills will be included in the new version. A report on integrating CT skills into the curriculum was published in 2018.[62] Most Spanish regions are developing initiatives to include Computational Thinking/ICT/digital activities in which CT skills are integrated. Primary schools are likely to incorporate such activities both as part of a cross-curricular approach and specifically within the subject of Mathematics. Some autonomous regions also have specific subjects in upper secondary education that deal with topics like robotics. In the development of the curriculum associated with the new Education Law issued in 2021,[63] digital competence is included as a cross-curriculum topic at all levels, with a direct reference to the development of CT skills. A comprehensive research report called "La Escuela de Pensamiento Computacional y su impacto en el aprendizaje"[64] and centred on the 2018-2019 school year is available, while an equivalent report focusing on the 2020-2021 school year is due to be published in early 2022.

60. Primary level curriculum (Grades 3 and 4): https://www.statpedu.sk/sk/svp/inovovany-statny-vzdelavaci-program/inovovany-svp-1.stupen-zs/matematika-praca-informaciami/; Lower secondary curriculum (Grades 5, 6, 7 and 8): https://www.statpedu.sk/sk/svp/inovovany-statny-vzdelavaci-program/inovovany-svp-2.stupen-zs/matematika-praca-informaciami/; https://www.statpedu.sk/sk/svp/inovovany-statny-vzdelavaci-program/
61. https://www.racunalnistvo-in-informatika-za-vse.si/racunalniskomisljenjezavse.html
62. http://code.intef.es/wp-content/uploads/2018/10/Ponencia-sobre-Pensamiento-Computacional.-Informe-Final.pdf
63. https://www.boe.es/diario_boe/txt.php?id=BOE-A-2020-17264
64. http://code.intef.es/wp-content/uploads/2019/12/Impacto_EscueladePensamientoComputacional_Curso2018-2019.pdf

Sweden

In March 2017, the Swedish government issued a revised curriculum with a definition of digital competence that featured programming. This new curriculum was officially enacted in Autumn 2018 but had already been voluntarily implemented by schools and teachers starting in 2017. Algorithms and programming are integrated in primary and lower secondary schools, mainly within three subjects: Technology, Mathematics and Civics.

Other European countries

Georgia

CT activities have been part of the curriculum in Georgia[65] since 2011. Computational (Algorithmic) Thinking is considered a core foundational concept behind Computing/Computer Science/Informatics. Currently, CT activities are integrated within other subjects as a compulsory element in primary school and as an elective element in secondary school. As part of a current curriculum revision,[66] Algorithmic Thinking will become mandatory from grades 2 to 6. The National Centre for Teachers' Professional Development[67] is actively planning the training and recruitment of new teachers to implement STEM subjects in schools, a measure that is relevant to CT skills.

Israel

In Israel, Computer Science is an elective subject offered from primary school (grades K4-K6) until school graduation, and is studied in most high schools. The computer science major for high schools (and junior high) is by far the biggest and fastest-growing programme in the country, encompassing more than 1,000 schools and involving more than 13,000 students annually. It provides core fundamentals of computer science theory and practice at the high school level. One of the programme's key principles is that core concepts are more important than teaching specific programming languages. Students are encouraged to learn how to program, but no particular programming language is prescribed.

65. http://ncp.ge/en/teqnologiebi/shesavali
66. http://ncp.ge/en/curriculum/competencies/digital-literacy
67. http://tpdc.gov.ge/eng/home/

Norway

CT skills have been part of the curriculum[68] in Norway since the 2020/2021 school year. They are integrated as a compulsory element within other subjects, including Maths, Science, Arts & Crafts, and Music. In Norway, the term Algorithmic Thinking (*Algoritmisk tenkning*) is mainly used, which is understood as a problem-solving method. The terms Algorithmic Thinking and Computational Thinking were first mentioned in policy documents in preparation for school pilot activities devoted to programming, which started in 2016. The adoption of CT skills and programming in the curriculum is grounded on the assumption that these provide an ideal foundation for fostering problem solving, logical skills, and digital competence. An overall evaluation of the curriculum's implementation, including its elements related to CT skills, has been initiated.

Russia

A new statutory curriculum for primary and general school education was introduced in 2021, including a new K7-K9 curriculum for Informatics. This new curriculum has a stronger focus on algorithms and programming and includes a section focusing on digital literacy. The term Algorithmic Thinking is used in Russia, and it is a fundamental component of the school subject Informatics,[69] which is compulsory at the secondary level. CT skills are integrated into this subject in secondary school, while they are part of other subjects at the primary level. In addition, various related activities are organised outside of formal education, such as Olympiads,[70] competitions and clubs. There are plans to include digital literacy in the mandatory minimum framework of assessed competences in Informatics (grades 1 to 11) and make suitable provisions for all schoolteachers.

Serbia

In Serbia,[71] "Informatics & Computer Science'" and "Technics & Technology" were introduced as compulsory subjects for students in lower secondary education (grades 5 to 8). In 2016, an elective course, "From Toys to Computers" (*Od igračke do računara*),[72] was introduced at the primary level from grades 1 to 4. In 2020, "Digital World"[73] was introduced as a compulsory subject at the primary school level. This includes topics like digital society, digital safety, and algorithmic thinking. The subjects mentioned above cover CT-related activities, which are also integrated into other subjects such as Mathematics, Philosophy and Civic Education. Back in 2013, the Serbian Government and Ministry of Education developed a strategy for educational development until 2020.[74] This strategy recognised technology and informatics as key skills for the future development of education. The new strategy for the upcoming period is in the process of development.

68. https://www.regjeringen.no/no/dokumenter/politisk-plattform/id2626036/#kunnskap
https://www.regjeringen.no/contentassets/dc02a65c18a7464db394766247e5f5fc/kd_framtid_fornyelse_digitalisering_nett.pdf
69. https://fgosreestr.ru/registry/primernaya-osnovnaya-obrazovatelnaya-programma-nachalnogo-obshhego-obrazovaniya-2/
https://fgosreestr.ru/registry/primernaya-osnovnaya-obrazovatelnaya-programma-srednego-obshhego-obrazovaniya/
https://fgosreestr.ru/registry/%d0%bf%d0%be%d0%be%d0%bf_%d0%be%d0%be%d0%be_06-02-2020/
70. Минпросвещения России (edu.gov.ru); Банк документов (edu.gov.ru)
71. Curriculum for the first grade of elementary school, including description of the new subject "Digital World" introduced in 2020 as new compulsory subject for lower elementary grades. This includes the topics digital society, digital safety and algorithmic thinking. Curriculum for the fifth and sixth grade of primary education (lower secondary level) for the subject Informatics and Computer Science (compulsory in higher grades of primary school), which includes the topics digital literacy, ICT, programming, computational thinking, and problem solving. Curriculum for the seventh and eighth grade of primary education (lower secondary level) for the subject Informatics and Computer Science (compulsory in higher grades of primary school), which includes the topics digital literacy, ICT, programming, Computational Thinking, and problem solving. Curriculum for gifted students in Computer Science Grammar School in Belgrade (upper secondary school level). In this school, Informatics, Algorithmic Thinking, Computation and Programming are one of the most important and taught courses. Curriculum for upper secondary schools. In these schools, "Informatics & Computer Science" is a compulsory course throughout all four years. In addition, Algorithmic Thinking is also mentioned in elective courses and philosophy. Curriculum for an elective course called "From toys to computers" (Od igračke do računara).
72. http://www.lugram.net/pdf/OD%20IGRACKE%20DO%20RACUNARA%20-%20nastavni%20planovi.pdf
73. https://zuov.gov.rs/wp-content/uploads/2020/08/pravinik-digitalni-svet.pdf
74. https://www.mpn.gov.rs/wp-content/uploads/2015/08/strategija_obrazovanja_do_2020.pdf

Switzerland

CT skills are part of "Media and Informatics", a compulsory subject implemented at the primary and secondary level, but they are also integrated across other subjects. Curricula[75] define learning outcomes for "Media and Informatics", some of which relate to the concept of CT. For example, students are expected to analyse simple problems, describe solution procedures and implement them in programs. Several initiatives to attract girls into Computer Science have been established, such as "Programming Workshops for Girls".[76]

UK-England

England was one of the first European countries to introduce the development of CT skills at all levels of school education. The core of the Computing programmes of study enacted in 2014 is computer science, in which students are taught the principles of information and computation, how digital systems work, and how to put this knowledge to use through programming. The Computing subject also includes digital literacy elements like using, expressing oneself and developing ideas through ICT at a level suitable for the future workplace and as active participants in a digital world. Created in November 2018 with £84m of government funding, the National Centre for Computing Education[77] developed a complete programme of study called "Teach Computing Curriculum".[78] This contains 500 hours of lesson plans, assessment exercises, practical exercises, student interaction activities, progression charts, concept charts, teacher guides, etc.

Non-European country Singapore

In 2020, a ten-hour enrichment programme devoted to learning Computational Thinking and coding through visual programming-based lessons[79] was made compulsory for students in upper primary education (grades 4-6). Generally, the term Computational Thinking is used. CT skills[80] are also taught in the Computing subject offered to grades 9 and 10 in some schools in Singapore, and also in the Computer Science subject at some schools for grades 11 and 12. As part of the National Digital Literacy Programme, some initiatives are also undertaken in the Mathematics curriculum to help develop and deepen CT skills at secondary levels.

75. *Plan d'études romand* (French-speaking cantons): https://www.plandetudes.ch/web/guest/education-numerique; Lehrplan21 (German-speaking cantons): https://v-fe.lehrplan.ch/index.php?code=b|10|0|2; *Piano di studio* (Italian-speaking canton): Piano di studio Tecnologie e media (ti.ch)
76. https://www.bfh.ch/de/ueber-die-bfh/service-beratung/mint/coders-lab/
77. https://teachcomputing.org/about
78. https://teachcomputing.org/curriculum
79. https://www.straitstimes.com/tech/coding-to-be-made-compulsory-for-all-upper-primary-pupils-next-year
80. 2021-o-level-computing-teaching-and-learning-syllabus.pdf (moe.gov.sg); Strengthening Digital Literacy | Committee of Supply 2020 (moe.gov.sg)

4.5 CT skills in upper secondary education curricula in Europe

At the upper secondary education level, 17 countries (AT, DK, EL, ES, FR, HR, IE, LT, PT, SK, SI – and CH, IL, NO, RS, RU, UK-ENG) offer Computational Thinking (or a related topic) as a separate subject. In nine countries (BE nl, DK, ES, FI, FR, HU, RO – and NO, RS), Computational Thinking (or a related topic) is taught within other subjects. In eight countries (AT, BE nl, EL, FI, HU, PL, SI – and CH), CT is taught as a cross-curricular theme.

In Belgium Flanders, Luxembourg, and Spain, the curriculum positioning of CT skills in both lower and upper secondary education depends in part on regional or school-level curricula.

Upper secondary education	AT	BEfr	BEnl	CY	DK	EL	ES	FI	FR	HR	HU	IE	IT	LT	LU	MT	PL	PT	RO	SK	SI	CH	GE	IL	NO	RU	RS
As part of a separate subject	✓				✓	✓	✓		✓	✓		✓						✓		✓	✓	✓		✓	✓	✓	✓
Within other subjects	✓		✓		✓		✓	✓	✓		✓							✓	✓						✓		✓
As a cross-curricular theme			✓			✓		✓									✓				✓	✓					
Regional or school policy			✓				✓						✓		✓												
No integration		×		×										×									×				
No data available																×											

Table 7. A look at CT skills integration in Europe's upper secondary school curricula
(Note: outside the scope of this study – provided solely for context)
Source: Authors' elaboration based on the study survey

4.6 CT in initial VET in compulsory education

Note: to explore CT skills integration also in iVET compulsory education streams and provide input for follow-up actions, this section reports results from the literature review, together with input collected during the online expert workshop and three written interviews with VET experts.

The world is going digital, and so is the labour market, with digital skills now a requirement in almost all occupations. Therefore, students in initial Vocational Education and Training (iVET) need to develop both job-specific skills and a solid digital skills component, including CT skills. As Yadav and colleagues put it (Yadav et al., 2017, p. 1064), "there is a need to pay attention to CT as part of the broader concept of digital literacy in vocational education and training, as otherwise adults with only professional qualification may not be well prepared for the working life in the twenty-first century".

Yadav et al. (2017) state that although the current policies and practices worldwide emphasise the need to integrate CT in curricula, "attention for CT in vocational education is mostly lacking" (p. 1062). To gather evidence on the integration of CT in compulsory-education level iVET settings, this study:

- searched the literature for relevant publications;
- devoted a discussion group to iVET during the expert workshop on CT skills;
- conducted three written interviews with VET experts from Germany, the Netherlands and Portugal.

Analysis of the collected data reveals that ICT/Computer Science/Informatics & Computation Thinking education demands are very strong. This is considered as a factor that can:

- raise students' employability and better prepare them for the (increasingly digital) job market;
- contribute to equity and social inclusion, especially by improving opportunities for girls and students from disadvantaged backgrounds;
- respond to demands for more (and better) IT and STEM professionals.

However, the **iVET sector is having difficulty responding appropriately** to these demands, with notable disparity across contexts and among the EU Member States. Some initial attempts to integrate CT skills, for instance, as part of digital literacy, are making progress but are proving insufficient.

Results from the survey of policy initiatives conducted for this study – reported in Section 2.2 – reveal that 12 countries (BE nl, EL, LU, IT, MT, PL, RO – and GE, IL, NO, RS, RU) have already integrated (or plan to integrate) CT skills or related concepts into their vocational education curricula in lower secondary education. By comparison, 15 countries have done so at upper secondary level (BE nl, EL, HR, HU, IT, LU, PL, RO, SI, SK – and CH, IL, NO, RS, RU).

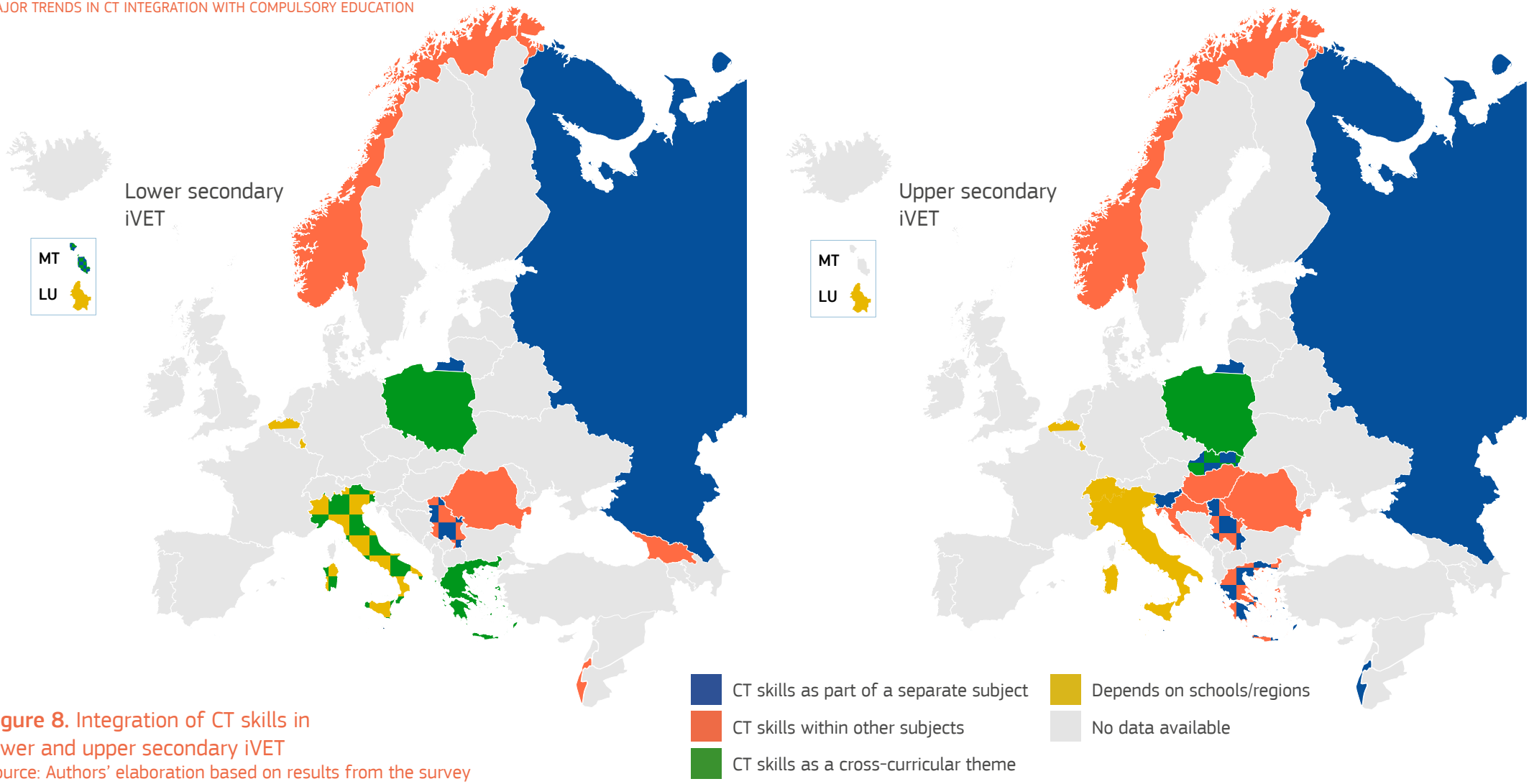


Figure 8. Integration of CT skills in lower and upper secondary iVET
Source: Authors' elaboration based on results from the survey

As depicted in Figure 8, the **various approaches for integrating CT skills in iVET curricula** (as a separate subject, within other subjects, as a cross-curricular topic, or as a combination of these options) mirror those observed in compulsory education generally. However, there is a general lack of clear policy guidelines about the space and time to be allocated in the curriculum to promote the development of CT skills and related concepts.

Analysis of data collected for the study indicates that, ideally, **CT skills could be incorporated into iVET both as a separate subject and in a cross-curricular manner**, but this position requires solid research backing. Unfortunately, **little research has been performed so far** in this area, and there are few indications of meaningful dialogue between the spheres of research and practice.

Strong intrinsic links between digital literacy and Computer Science skills make CT skills a real-world and domain-general competence (e.g., CT *via* and *for* problem solving), something that is – or should be – related to diverse subjects and disciplines. However, CT is under-addressed in iVET curricula and does not have a high profile in this sphere of compulsory education. Programming is often adopted in iVET education programmes to develop CT skills and related concepts. Indeed, there is often an overemphasis on functional CT skills in specific iVET tracks (both by educational authorities and external entities) rather than promoting a broader sense of CT as a skill set for all iVET students. So there is a **need to integrate more abstract CT concepts in ICT/CS/Informatics and other courses within all iVET tracks**.

However, integrating CT and related concepts in iVET curricula is not a straightforward endeavour. Analysis of the data collected for this study reveals that teachers' and trainers' competences are crucial for the successful integration of CT in iVET. The OECD (2021) emphasises that VET teachers tend not to use learner-centred pedagogies or practical, real-life situations for promoting learning, despite the applied nature of vocational education. Data from six OECD countries and regions with available TALIS data reveal that only 36% of VET teachers stimulate their students by proposing open-ended tasks, and only just over half challenge them to solve complex problems. This tendency may represent a barrier to introducing CT skills in iVET settings, as CT actually entails problem solving and tackling open-ended tasks. Therefore, significant challenges for integrating CT skills in iVET include providing upskilling through professional development programmes to catch up with new developments. There is a need for massive investment in proper initial education, in-service training, and other upskilling opportunities, as well as for helping companies that provide iVET programmes – especially Small & Medium Enterprises (SMEs) – to integrate CT skills into their training programmes. In addition, there is a need to provide teachers from other disciplinary areas in iVET with tailor-made professional development programmes that help them develop the skills necessary to incorporate CT-related activities into their teaching. Special vocational training platforms ought to be made available to exchange training concepts, modules, and training content.

There is also a need to address **the shortage of qualified iVET teachers**.[81] For instance, Forlizzi and colleagues (2018) report that Italy and other countries lack teachers who have sufficiently familiarity with the basic concepts of informatics. The authors point out that "even in vocational schools it is common to find teachers with a poor background in the [computing] field and, given the current state of the recruitment process, we can hardly hope that the situation will improve in the next few years" (p. 151).

One key message from the expert interviews conducted for this study is the need to systematically develop the didactics of CT/Informatics and quality standards for teaching material and training. Another finding is that it would be beneficial to create standardised competence frameworks and provide examples of best practices, concrete learning scenarios, and quality content. Doing so would help vocational school teachers and trainers from companies that provide iVET to systemically implement CT skills in their teaching. It is essential that such frameworks, learning scenarios, and content cater for learners' heterogeneous backgrounds and competences, and for the diverse structures and capacities of the companies – from SMEs to large companies – providing iVET.

In addition, there is a need for private-public collaboration between iVET institutions and businesses (of different nature) to jointly develop curricula, educational innovation, and training opportunities. Schools, teachers & trainers, and companies all need to be more closely involved in bottom-up innovations that can be progressively scaled up to the system level. For instance, Crick (2017) calls for an apprentice model for teaching programming, whereby students learn their craft from a master, given that programming's inherent complexity makes it particularly challenging to teach. This is an interesting proposal and may be relevant for integrating CT both in iVET and general education settings.

81. "Qualified teacher" refers to any teacher with the competence to teach CS - and help learners develop CT skills – in a sound and effective manner, irrespective of how that competence was acquired. It does not assume the teacher possessing official certification of that competence, or having followed a certified course of study to achieve it, although that may indeed be the case.

Another critical aspect for integrating CT in iVET in many countries is that qualification for the labour market derives from upper secondary vocational education. Lower secondary vocational education often has a more preparatory function. Therefore, the quality of continuous learning trajectories and pathways from lower to upper secondary, further, and continuous VET is paramount. Vocational institutions require all the resources and infrastructure necessary to integrate CT skills and related concepts.

Finally, evidence from this study reveals minimal provisions for equity issues and gender balance in integrating CT in the iVET curricula. VET for ICT jobs "might themselves have developed gender- and class-biased occupations, with girls being encouraged to explore areas of «soft» ICT skills more, so that they end up with what is regarded as employment with lower-level skills" (Kirkup as cited in Passey, 2017; p. 431). Box 2 provides a synthesis of follow-up actions for integrating CT skills in iVET settings.

Box 2 Input on follow-up actions for integrating CT skills in iVET settings

The demand for integration of CT skills in iVET settings within compulsory education is high, and iVET institutions face similar challenges as those in general education do. However, there are some characteristics that are unique to iVET, such as the role that trainers and businesses of different nature can play for integrating CT skills in training and apprenticeship programmes. Some input for follow up actions in iVET settings is provided below.

- Devote attention to CT as part of the broader concept of digital literacy.
- Integrate more abstract CT concepts in computing (e.g., Informatics) and other courses within all iVET tracks.
- Provide clear policy guidelines about the space and time to be allocated in the curriculum to promoting the development of CT skills and related concepts.
- Support research and establish a meaningful dialogue between the spheres of research and practice.
- Explore the effectiveness of an apprentice model for teaching programming, where students learn their craft from a master in the workplace.
- Systematically develop the didactics of CT/Informatics and quality standards for teaching material and training.
- Create standardised competence frameworks, and provide best practices, concrete learning scenarios, and quality content for the integration of CT skills.
- Promote public-private collaboration between iVET institutions and businesses (of different nature) for joint development of curricula, educational innovation, and training opportunities.
- Provide initial education and upskilling through professional development programmes so prospective and in-service educators can catch up with new developments.
- Help companies that provide iVET programmes – especially SMEs – to integrate CT skills into their training programmes.
- Ensure continuous learning trajectories and pathways from lower to upper secondary, further, and continuous VET.
- Provide iVET institutions with the resources and infrastructure necessary to integrate CT skills and related concepts.
- Address equity issues and gender balance in integrating CT skills in iVET curricula.

5

Approaches to CT teaching, learning and assessment

5.1 In-depth case studies in nine European countries

To understand better what is being taught and why it is being taught, this section discusses the results of nine in-depth studies: seven in EU Member States (FI, FR, HR, LT, PL, SE, SK), and two in other European countries (NO, UK-ENG). To strengthen the reliability of the outcomes, a multiple-case study approach was adopted (see Section 2 – Methodology).

The data for each case study were drawn from analysis of their national curricula and associated support provisions, and also from a total of 38 interviews with school leaders, teachers in charge of CT integration[82], experts, and policy makers from the nine different countries. In addition, ten focus groups were conducted, each involving a cohort of five students aged between nine and 14 years old. The Case Study schools were recommended by Ministries of Education or identified by national experts in the field; consequently, they are generally quite advanced in implementing CT skills compared to other schools in their country. The range of different sources involved in the selected case studies made it possible to capture both policy and implementation perspectives. All these case studies fall within compulsory education. The rationale for this decision lies (i) in the nature of compulsory education, which has a broad mandate to provide all students with key competences, and (ii) in the average ten-year time span of compulsory education in European countries, generally involving students aged six to 16 (European Commission/EACEA/Eurydice, 2020). For sound cross-case comparisons to be made, the cases were all carefully selected to ensure that both similar and contrasting results could emerge from the overall case set (Yin, 2003). For this reason, the nine individual cases in the multiple-case study investigation (see Figure 10) were selected according to the following criteria:

- development of CT core skills (e.g., programming, algorithmic thinking) as a compulsory part of the statutory curriculum or guidelines;
- approach to CT integration in the statutory curriculum or guidelines, (i.e., CT skills as a specific subject; as part of other subjects; as a cross-curriculum theme) (Bocconi et al., 2016);
- variety in geographical coverage across Europe;
- falling within compulsory education (i.e., primary – ISCED 1, lower secondary – ISCED 2);
- maturity level (case initiative implemented for at least a two-year duration, since 2015);
- backing with appropriate policy support measures (e.g., subsidiarity, Professional Development for teacher upskilling).

Given the different organisation and duration of primary (ISCED 1) and lower secondary education (ISCED 2) in the three Multiple-Case Studies (MCS), the analyses focused on specific grades levels (see Table 7): i.e., MCS1 focused on grades 3-4 (LT, NO, SK); MCS2 focused on grades 5-9 (HR, PL, UK-ENG); and MCS3 focused on grades 7-9 (FI, FR, SE).

82. In the multiple-case study regarding CT *skills addressed within other subjects* (e.g., *Maths and Tech*), for each of the three country cases involved (FI, FR, SE) interviews were held with two teachers (one from Maths and one from Technology), as well as with one expert in CT integration, who shed light on synergies between the two subject areas involved.

Nine in-depth case studies

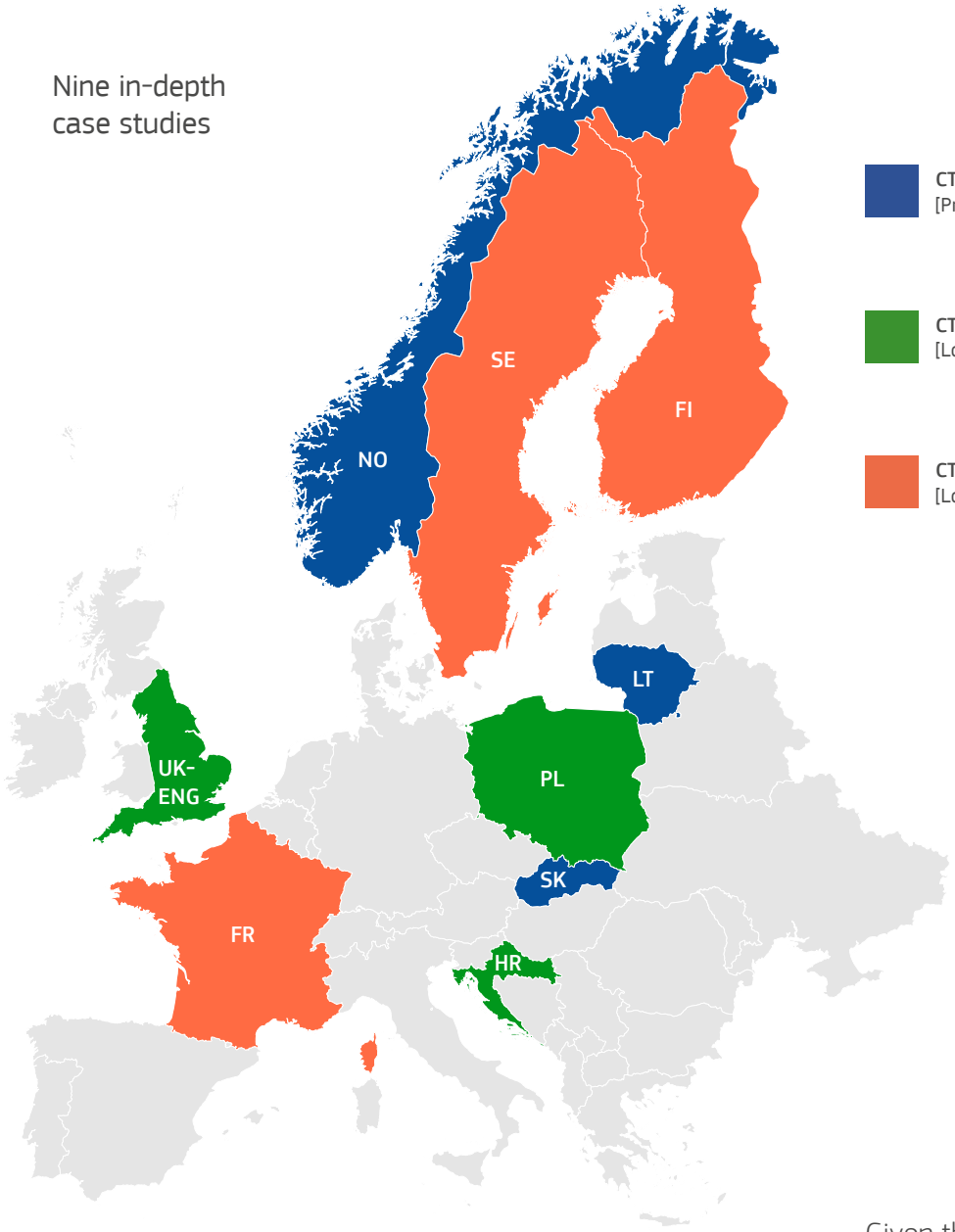


Figure 9. Distribution of the nine case studies in European countries and their thematic groupings
Source: Authors' elaboration

CT SKILLS AS *CROSS-CURRICULAR THEME*
[Primary Education - ISCED 1]

*Multiple-Case 1 (**Lithuania**, **Norway** and **Slovakia**)*
Scope: comparing approaches to integrating basic CS concepts
1. as a *cross-curricular theme* (LT and NO) and
2. as part of a Computing/Informatics subject from grade 4 (8 y/o students) (SK)

CT SKILLS AS *PART OF A SEPARATE SUBJECT*
[Lower Secondary - ISCED 2]

*Multiple-Case 2 (**Croatia**, **Poland** and **UK-England**)*
Scope: comparing approaches to developing CT skills in contexts:
1. with a long-standing tradition in Computer Science (HR and PL) and
2. moving from ICT to Computing subject (UK-ENG)

CT SKILLS *WITHIN OTHER SUBJECTS*
[Lower Secondary - ISCED 2]

*Multiple-Case 3 (**Finland**, **France** and **Sweden**)*
Scope: comparing approaches where:
1. CT skills are developed within Maths & Technology subjects (FI and FR) and
2. CT skills are component of Maths, Technology and Civic subjects as part of an evolving definition of digital competence (SE)

GRADES

AGE	ISCED 1			ISCED 2					
	LT	NO	SK	HR	PL	UK-ENG	FI	FR	SE
6	1	1	1						
7	2	2	2						
8	3	3	3						
9	4	4	4						
10		5							
11		6		5	5	7		6	
12		7		6	6	8		7	
13				7	7	9	7	8	7
14				8	8		8	9	8
15							9		9

Table 8. Grades analysed in the nine countries examined in the in-depth case studies
Source: Authors' elaboration based on Eurydice data[83]

Given the different organisation and duration of primary (ISCED 1) and lower secondary education (ISCED 2) in the nine European countries, the analyses focused on specific grades levels. Table 8 shows the grades levels involved in each case (within red dashes). Concerning the three primary education (ISCED 1) cases analysed, the in-depth case studies focused on grade levels 3-4 in Lithuania, Norway and Slovakia. In lower secondary education (ISCED 2), the analysis focused on grade levels 5 to 9 in HR, Pl and UK-ENG, and from grade levels 7 to 9 in FI, FR and SE.

83. https://eacea.ec.europa.eu/national-policies/eurydice/national-description_en

5.2 What is taught: core Computer Science contents in the nine case studies' curricula

Analysis of the nine curricula revealed two main interrelated strands: one dealing with basic Computer Science (CS) concepts and another regarding Digital Competence/Digital Literacy elements. Both strands are present in the five curricula where CS concepts are integrated as a separate subject (i.e., in Croatia, Lithuania, Poland, Slovakia and UK-England). In the other four curricula (i.e., in Finland, France, Norway and Sweden), where CT skills are developed *within other subjects*, CS concepts are predominantly dealt with in Maths. In Finland, France and Sweden, both strands are addressed in Technology. In Norway, both strands are dealt within the STEAM subjects Science, Arts & Crafts, and Music (detailed summaries of the Multiple Case Studies can be found in Annex 4).

Therefore, in the nine case studies, regardless of which integration approach has been adopted, the **two major strands mentioned above (basic CS concepts and Digital Competence/Digital Literacy elements) are present in the subjects where CT skills are developed**.

In the first strand, basic CS concepts mainly revolve around the relationships between **algorithms** and **programming**. As depicted in Figure 10, analysis of the nine curricula revealed a set of concepts related to algorithms (including abstraction, the notion of algorithms, algorithmic problem-solving, and logical reasoning). Concepts identified in relation to programming include the notion of programming, variable, conditional, loops, sequence, creating a program, and debugging. In addition, concepts concerning the relationship between algorithms and programming include design, decomposition, and pattern recognition for developing a computable solution to a given problem. The concepts in blue and orange shown in the figure in the right are present both in primary and lower secondary curricula, where they are addressed at different levels of complexity. In lower secondary curricula, a number of more complex concepts are addressed (those labelled in black), such as data structure, procedures, Boolean logic, functions, event, and evaluation.[84] The full comparison of basic CS concepts from the nine curricula is available in Annex 5.

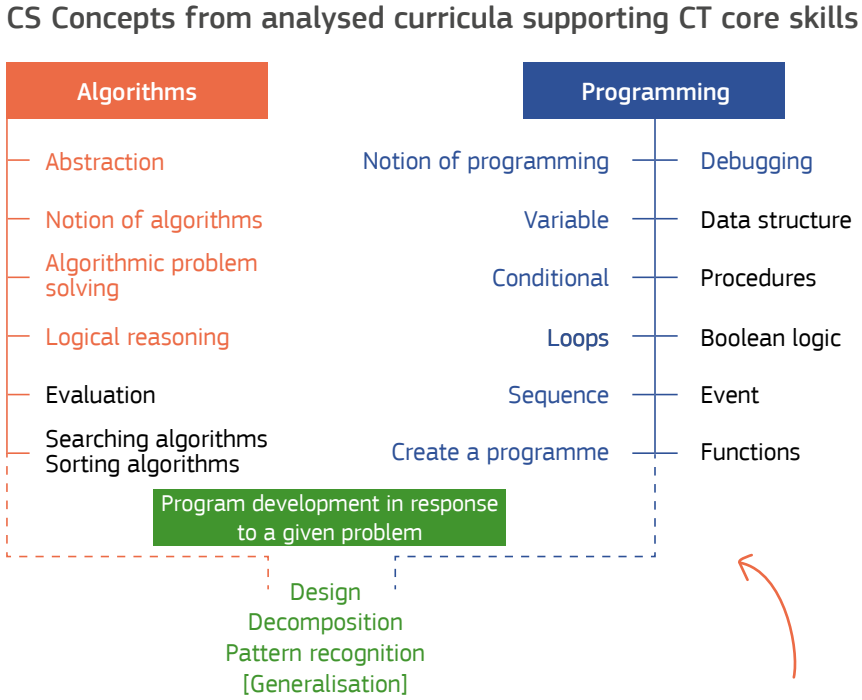
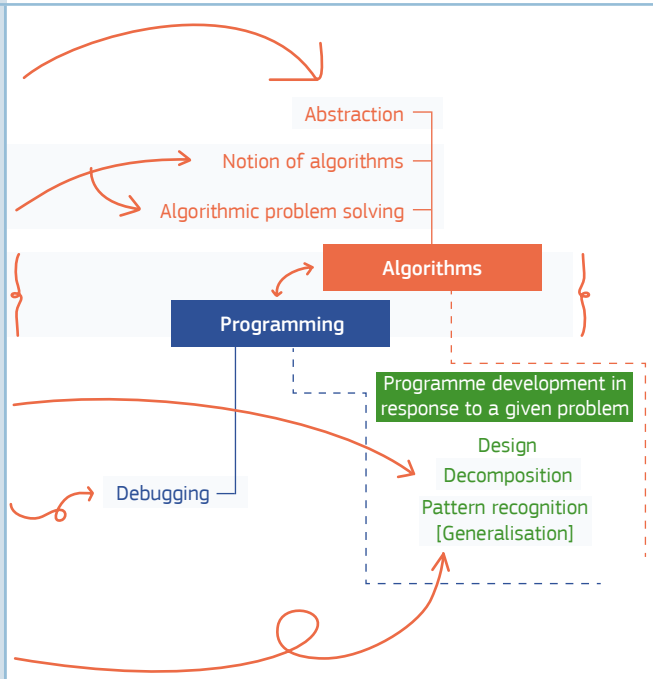


Figure 10. Set of basic Computer Science concepts from nine analysed curricula supporting core CT skills
Source: Authors' elaboration

This set of basic CS concepts that emerged from analysis of the nine curricula provides the grounds for developing CT skills like abstraction, algorithmic thinking, automation, decomposition, debugging and generalisation. This set of core CT skills forms an integral part of the working definition of CT proposed in the earlier generalization. This set of core CT skills forms an integral part of the working definition of CT proposed in the earlier 2016 CompuThink study (Bocconi et al., 2016).[85] Figure 11 below maps the basic CS concepts that emerged from the analysed curricula against the set of CT skills.

84. "Event" is a concept present only in the French curriculum, "evaluation" only in the UK-English curriculum.
85. The set of core CT skills identified in the 2016 CompuThink study can be found, with minor variations, in a number of other CT-related initiatives, including work by Grover & Pea (2013); Csizmadia et al. (2015); K-12 Computer Science Framework (2016); ICILS CT Framework (Fraillon et al., 2019); PISA 2022 Maths Framework (OECD, 2018).

CT Skill	Definition
Abstraction	Abstraction is the process of making an artefact more unserstable through reducing the unnecessary detail. The skill in abstraction is in choosing the right detail to hide so that the problem becomes easier, without losing anything that is important. A key part of it is in choosing a good representation of a system. Different representations make different things easy to do (Csizmadia et al., 2015, p. 7).
Algorithmic thinking	Algorithmic thinking is a way of getting to a solution through a clear definition of the steps (Csizmadia et al., 2015, p. 7).
Automation	Automation is a labour saving process in which a computer is instructed to execute a set of repetitive tasks quickly and efficiently compared to the processing power of a human. In this light, computer programs are "automations of abstractions (Lee, 2011, p. 33).
Decomposition	Decomposition is a way of thinking about artefacts in terms of their component parts. The parts can then be understood, solved, developed and evaluated separately. This makes complex problems easier to solve, novel situations better understood and large systems easier to design (Csizmadia et al., 2015, p. 8).
Debugging	Debugging is the systematic application of analysis and evaluation using skills such as testing, tracing, and logical thinking to predict and verify outcomes (Csizmadia et al., 2015, p. 9).
Generalisation	Generalisation is associated with identifying patterns, similarities and connections, and exploiting those features. It is a way of quickly solving new problems based on previous solutions to problems, and building on prior experience. Asking questions such as *Is it similar to a problem I've already solved?* and *How is it different?* are important here, as is the process of recognising patterns both in the data being used and the processes/strategies being used. Algorithms that solve some specific problems can be adapted to solve a whole class of similar problems (Csizmadia et al., 2015, p. 8).



CORE CT SKILLS FROM 2016 EC COMPUTHINK REPORT

BASIC COMPUTER SCIENCE CONCEPTS FROM NINE CURRICULA

Figure 11. Mapping of basic CS concepts that emerged from the nine case-study curricula against core CT skills
Source: Authors' elaboration

Notably, algorithms and programming are also among the fundamental concepts of Computer Science (CS). These two concepts, which were the first to emerge in the development of CS, also provide basic grounding to all students. This is consistent with the notion that the historical progression of concepts within a discipline may reflect a similar cognitive progression in learners (Denning & Tedre, 2021; Knuth, 1985). In a long-term perspective, integration of elements of machine learning and Artificial Intelligence should be considered at lower secondary level, drawing on the latest research on how to address such topics in an age-appropriate way (Kahn & Winters, 2021).

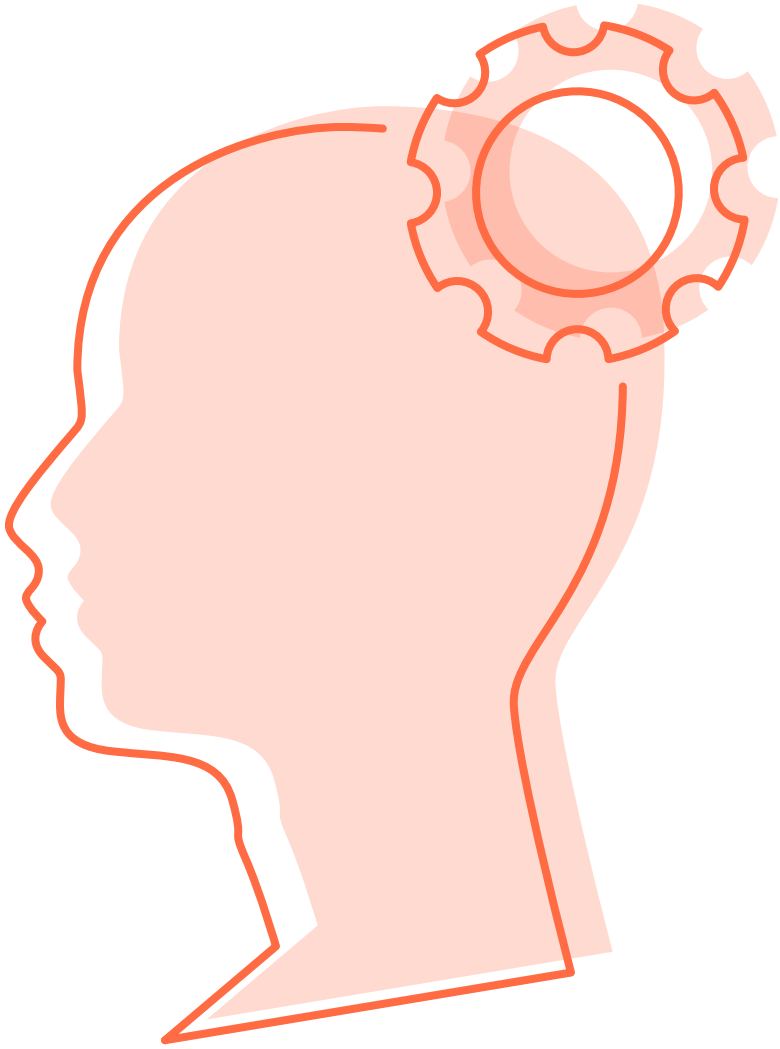
The second strand of contents dealing with digital competence/digital literacy covers the following common set of elements:

- information literacy;
- data analysis and management;
- creating, re-using, revising, and re-purposing digital resources;
- safe, respectful, responsible and secure use of technology;
- problem solving.

Why Computational Thinking skills are integrated in case study curricula

A common objective across the analysed curricula is to help students develop basic Computer Science understanding along with their digital competence. As is generally acknowledged, digital competence enables students to be critical and active participants in a digital world, with a broad skill set to develop ideas and express oneself effectively and appropriately through digital technology (Vuorikari et al., 2016). Hence, complementing digital competence with CT skills should help prepare young people to effectively deal with the demands of the increasingly complex digital world and workplace. This rationale also emerged during the interviews that this study conducted with policy makers, school leaders, experts, and teachers.

As outlined above, in the nine case studies, basic CS concepts underpin core CT skills and support and complement the development of digital competence. This position is supported by Hsu et al., (2019), who argue that introducing CT as compulsory for all students is based on the premise that "CT is a foundational skill that all students should have to be digitally competent and active participants in a world where computing is pervasive" (p. 268).



5.3 Contextual factors affecting how Computational Thinking is implemented in case studies

CT implementation in the nine case-study countries is affected by a number of contextual factors. In the following, three main factors that emerged are examined.

The first contextual factor regards the type of CT integration approach[86] adopted at a specific education level. Analysis of the three case studies at primary level (Lithuania, Norway and Slovakia) revealed that a mixture of integration approaches was adopted. One of the peculiarities of primary education is that teachers cover several subjects in their practice without necessarily having specialised subject-area expertise in each one. This condition favours a cross-curricular theme approach. However, this study revealed that this cross-curricular approach is always coupled with integration as *a separate subject* (Approach 2) or *within other subjects* (Approach 3). The rational that interviewees gave for this was that basic CS concepts first need to be developed in the context of a specific subject, and only later can they be applied in other subjects. For example, in Lithuania and Slovakia, CT skills are first developed within Informatics and are then applied across disciplines as well. The same applies in Norway, except that CT skills are first developed as part of Maths. This is in line with, and supports, the development of digital competence as a cross-curricular theme in primary school.

Analysis of the six case studies conducted at lower secondary level confirmed that two integration approaches are applied: *as part of a separate subject* (Croatia, Poland, UK-England) and *within other subjects* (Finland, France, Sweden). In Croatia, Poland and UK-England, basic CS concepts are addressed and coupled with digital competence elements as an integral part of the Computing/Informatics subject at lower secondary level.

In Finland, France and Sweden, basic CS concepts are addressed within Maths and Technology. This approach exploits specific relationships between CS and Maths & Technology subjects. Specifically, the more abstract components of CS are more commonly developed in Maths, whereas physical computing (e.g., robots, programmable controlling devices) is more prominent in Tech, together with elements of digital competence.

The second contextual factor regards the amount of time dedicated to developing CT skills. At lower secondary level, where basic CS concepts (i.e., algorithm and programming) are addressed *within other subjects* (Maths & Tech), the amount of time dedicated to developing CT skills is left up to the individual teacher; national curricula guidelines only define the total number of hours allocated to different subjects. Hence, how much time the teacher devotes to developing CT skills depends on two interrelated aspects: (i) the overall time allocated to the subjects involved, and (ii) the content load to be addressed within those subjects.

In Finland, France and Sweden, during the three-year education span considered in the analysis (grades 7-9), Maths occupies the second largest space in the curriculum after language, covering approximately 400 hours across the three years: specifically, 418 hours in Finland, 378 hours in France, and 400 hours in Sweden. Regarding the overall breadth of subject contents to be covered, in Maths the situation varies significantly across the three cases:

- in the Finnish curriculum, Algorithmic Thinking is one of the 20 objectives stated under Maths;
- in France, Algorithms & Programming is one of the six content areas in Maths;
- in Sweden, programming is included as a sub-topic under the core contents of Algebra and Maths problem-solving.

86. As illustrated in Section 4, three different integration approaches for the introduction of basic CS concepts are adopted: (1) *as a cross-curricular theme*; (2) *as part of a separate subject*; (3) *within other subjects*.

A similar (and even more challenging) situation applies in the Technology subject, which occupies less curriculum time than Maths (76 in hours Finland, 162 in France, and 88 in Sweden) and where both strands (basic CS concepts and Digital Competence/Digital Literacy elements) are addressed. During the interviews conducted in this study, teachers in Finland and France pointed out that the average time allocated to algorithms and programming in Maths & Technology is **less than one hour per week**.

In Poland, the **Informatics** subject is usually taught **one hour per week**, while in Croatia, Informatics occupies two hours per week in grades 5 and 6. At primary level, teachers cover several subjects and have a higher level of flexibility in deciding the breath of curriculum contents to cover within a given subject. So, in summary, on average more than one hour per week (in each class) is dedicated to developing CT skills (e.g., in Norway, where CT skills are also developed within STEAM subjects). Hence, regarding the contextual factor of time, the aspect for attention is when CT is addressed *within other subjects* at lower secondary level.

The last contextual factor regards cooperation among teachers of the subjects in which CT skills are developed. At lower secondary level, when Algorithms and Programming are addressed *within other subjects*, dialogue and cooperation between Maths and Technology teachers is supposed to occur at curricula level, although if this is not adequately supported and encouraged, it is unlikely to happen.

As emerged from the interviews with Maths and Technology teachers in Finland, France and Sweden, one of the main challenges is to integrate the different approaches underpinning the practices adopted for teaching basic CS concepts in these two disciplines: abstract conceptualisation of programming in Maths and its more concrete and tangible application in Technology. This also emerged during the focus group with students, who expressed their preference for the activities conducted in Technology, e.g., programming robots, constructing a level crossing model with sensors to understand how that technology works. They considered these to be more engaging and less difficult activities compared to Maths tasks like using Scratch to solve open maths problems or constructing video games.

This also reflects different traditions in the pedagogies of Maths as opposed to Technology. Indeed, the French expert interviewed in the study emphasised that this difference derives from two distinct and deeply rooted

traditions in Maths and Technology education research: the notion of “didactic transposition” introduced by Yves Chevallard (1992) for mathematics, namely that the knowledge taught at school is derived from scholarly knowledge; and the “social practices of reference” proposed by Martinand for technology, whereby the formulation of teaching content cannot be limited to academic knowledge but should also consider social practices related to the application of that knowledge (Martinand, 2014).

Hence, when students’ core CT skills at lower secondary level are developed within other subjects like Maths and Technology, this contextual factor is of relevance for implementing quality computing skills. To this end, case study analysis revealed the need to support integration of CT learning across Maths and Technology. It also highlighted that research and monitoring mechanisms for CT-related initiatives in curricula ought to be created to specifically investigate how the two different approaches (CT within a Computing/Informatics subject versus CT addressed within other subjects like Maths and Technology) respectively impact on students’ development of core CT skills.

5.4 Pedagogical approaches

Overall, teachers select and combine educational methods and tools primarily based on leaner age-appropriateness, often exploiting pedagogical approaches commonly applied in other learning contexts as well.

Primary teachers (in LT, NO and SK) refer to approaches including **playful learning, learning by doing, learning from mistakes,** and **working in small groups**. Specifically, they highlight how students are introduced to basic Computer Science (CS) concepts using hands-on, playful activities with programmable robots and visual block environments. When developing their own projects, learners start by giving physical and/or virtual objects sequences of instructions to perform. By controlling robots or constructing programmes through a sequence of instructions, learners gradually move from being passive users of technology to being active constructors of digital objects. Primary teachers also report how teacher-student and student-to-student discussions are encouraged to promote **peer learning approaches**, e.g., the teacher encourages more eager learners to help any classmates who have got stuck somewhere.

At lower secondary level, CS curricula focus on aspects like problem-solving and logical thinking skills. These lend themselves to pedagogical approaches that promote student autonomy/agency, such as **personalised learning, project-based learning, game-based approaches, collaborative learning and pair-programming,** as well as **individual-oriented learning approaches**. Primary and lower secondary teachers also emphasise **the value of debugging** for creating a culture of learning-through-error, encouraging students to re-evaluate the significance of errors as a means for strengthening their expertise. All six lower secondary curricula analysed in the countries (FI, FR, HR, PL, SE, UK-ENG) seem to share a certain degree of openness, granting teachers autonomy in how to handle implementation. At the same time, this openness is scaffolded with step-by-step guidance, teaching resources, and examples for teachers who need more support.

What seems to be at the core of successful approaches to CS/Informatics curricula is enabling students to work on real-life problems and encouraging them to create something of their own, such as programs, applications, animations and so on. Another key aspect is promoting an iterative approach, whereby students are encouraged to check their ongoing production work, identify and correct any errors, and repeat this process in cyclical fashion.

An active learning approach is generally encouraged through student construction of simple programs. In the interviews with students and teachers, **peer collaboration, pair-programming** and **individual-oriented learning approaches** were all mentioned as pedagogical methods employed. In programming activities, particular attention is placed on generalising solutions as an introduction to algorithmic thinking. In Technology, a **project-based approach** is commonly used as a way to engage students in motivating activities.

Results from this study's literature review echo the case study findings concerning pedagogical approaches. Regarding *pair programming*, Campe at al. (2020) propose the rotating of roles in student collaboration activities. During pair programming, the two students alternate between the *driver* role (taking control of the keyboard and guiding program implementation) and the *navigator* role (providing input, looking out for mistakes, and accessing resources). Periodically alternating these roles about every 15 minutes allows both students to assume an active and a reflective attitude during the programming process. This approach calls for suitable design and structuring of learners' programming activities that teachers embed into a collaborative framework, where structured interactions play a crucial role and contribute to enriching the learning process.

Grover and Floyd (2020) propose the Question and Inquiry approach to fully exploit the potential and benefits of **teacher-student dialogue** for learning and problem solving in CT skills development. This is based on Polya's problem-solving process, i.e., understanding the problem, making a plan, executing the plan, and then looking back and reflecting. The authors define a taxonomy of questions to help teachers prompt students' engagement with programming concepts and procedures. They also advise on how teachers might respond to students' questions during the programming process. Recognising the type of questions to pose is crucial to the teacher's efforts to effectively support students in progressing through the inquiry process without impeding their independence and creative problem-solving.

Moreover, to help students deal with increasing complexity in programming activities, a number of pedagogical approaches for scaffolding the development of programming skills have been developed both for primary and lower secondary school. The main characteristics of these approaches is to start with sample programs exemplifying one or more programming concepts; then students start investigating and move on to 'modify' and finally use the concepts demonstrated for creating their own programs. Boxes 3 and 4 describe two examples of these approaches in action: TIPP&SEE and PRIMM.

Box 3 TIPP&SEE scaffolding strategy for developing primary pupils' programming skills

TIPP and SEE is a structured scaffolding strategy to help students learn programming by using and modifying existing Scratch projects to create their own. Scratch projects are organised around sprites, each having scripts that are executed in response to events. The teacher proposes a project that uses a particular programming concept that students investigate so they can understand how it works. They do this by running the project, looking inside to examine sprite scripts, and exploring what happens after modifying the scripts. This strategy is a specialisation of the Use-Modify-Create approach tailored to the Scratch programming environment, which features a platform for users to publish and share their projects. Each Scratch project has a title and instructions on how to use it, so the first part of the strategy guides the students to get a TIPP from the project web page, namely: Does the Title tell you something about the project?; What do the Instructions tell you to do?; What is the Purpose of this activity?; Play with the project, run the code and observe what happens. In the second part, the learner is guided to SEE inside the project to read and comprehend the code. The suggested roadmap (SEE) is to start by clicking on the Sprite to examine, then look at the Events blocks and the associated scripts (the teacher provides the students with worksheets that set questions and tasks to perform). Finally, the worksheet has suggestions to Explore by modifying the scripts. TIPP and SEE is described on the developers' website (https://www.canonlab.org/) along with the Scratch Encore Curriculum (https://www.canonlab.org/scratch-encore) for introducing Computer Science to primary school students.

Box 4 Predict, Run, Investigate, Modify, Make (PRIMM) approach for scaffolding lower secondary students' programming skills

Predict, Run, Investigate, Modify, Make (PRIMM) is a structured sequence of activities for teaching programming concepts to learners. It starts with the presentation of a program that students are asked to read. Then, working in pairs, they make a written prediction of what the program does. When everyone has made their predictions, the code is run online so students can quickly check their predictions and possibly discuss them with the class. To support reflection on the predictions and the actual results from running the program, the teacher provides a set of activities (e.g., code tracing, guided explorations, answering questions, debugging) to help students investigate the structure of the program. To continue exploration of the proposed program, the students are challenged to modify it, changing its functionality at progressive levels of complexity. During this phase, when moving from using somebody else's program to modifying it, there is a transfer of ownership from "not mine" to "partly mine". Once the students are confident in modifying the program, they can make their own versions, reusing the same/modified structures in a new program. The PRIMM approach builds upon and combines results from several research studies into supporting students in aspects of learning to program: reading and tracing code, adapting existing code, and moving between different levels of abstraction. The PRIMM portal (https://primmportal.com) provides a set of lesson plans to scaffold lower secondary students learning to program in Python. The portal also documents the research work behind the approach's development.

Box 5 Example of a spiral approach for addressing the relationship between algorithms and programming in practice

This example has been developed by the authors of this report to depict how the notion of algorithms can be introduced to younger students, starting from a recipe-like sequence of instructions and progressing to the application of shuffling algorithms. Although this example appears simple, it is not simplistic, depending on the conditions applied. It also provides an example of how to address age-appropriateness in a spiral approach.

Children develop a story or a game including elements that change randomly. Even at a young age (5-7) students have an intuitive idea of "randomness" derived from the use of random generators in games (e.g., tossing of dice, coins, lottery draws). Since programming environments provide random primitives (e.g., extracting a random number between one and six), learners can insert random elements in their programs. This does not imply student mastery of random concepts. While in primary education more playful approaches are used, at lower secondary level design aspects like reflect before programming become prominent. With students aged 9-10, the random element can be an itinerary. This implies instructing the computer to generate a random sequence without repetitions. To do so, students need to develop a shuffle algorithm. This can be approached with students using the intuitive idea of discarding repetitions in random extractions, e.g., taking note of elements already extracted. However, automated implementation requires a degree of scaffolding from the teacher to implement the discarding-repetitions mechanism by writing a program. Developing shuffling algorithms which can be efficiently executed by a computer (i.e., optimising time) and that are genuinely random is part of more advanced algorithm studies in Computer Science.

5.5 What progression occurs in relation to students' age

Note: to explore progression through curricula, this section concentrates on results from the literature review, since the nine case studies analysed only deal with specific grade levels (see **Table 8**. Grades analysed in the nine countries examined in the in-depth case studies).

In the initial years of primary school (students aged 5-7), learners' cognitive development with respect to the different facets of Computational Thinking (CT) skills development is best addressed through work on spatial skills and physical computing, e.g., robots. In all the three case studies conducted at primary level, a programming environment is adopted as a creative space for developing storytelling, use of interactive multimedia or connection with other topics taught. This is intended to help learners around the age of seven to further develop their working memory, dexterity skills, and language abilities. Since the cognitive ability of young learners of different ages varies significantly, methods for CT skills cultivation and the criteria for content development/selection should vary accordingly (Hsu et al., 2018). It is possible to either overwhelm or underwhelm students with content they feel they are either too young or too old for, leading to disenchantment with computing and computational thinking (Australian Computing Academy, 2019). The authors emphasise that students at different grade levels performed better on different concepts (Tikva & Tambouris, 2021). According to Rijke et al. (2018, p. 86) "Older students were found to do better on the abstraction task than students in the youngest age group. [...] No differences in performance on the decomposition task were found, either between age groups, or between males and females."

Forlizzi et al. (2018) **identify three main learning stages** in the CT skills area, stating what learners should be able to understand and perform from early ages onward:

1. In the first stage (primary school) learners are encouraged to ask questions, as well as to discover some basic ideas of informatics/computer science in their everyday life and explore these. They can be engaged either in plugged or unplugged activities;
2. In the second stage (lower secondary school) students are expected to grow in autonomy. They are expected to learn more about the organisation of data and the algorithm concept, to develop abstract thinking, and to acquire new specific and cross-disciplinary skills. Programming tasks can play a key role in this respect;
3. By the end of third stage (upper secondary school) students should be able to model problems and design algorithms. Abstraction, organisation, and accuracy are essential traits of the problem-solving approach in the informatics/computer science field; they foster the development of critical thinking and provide helpful keys to master complexity.

Hromkovič and Lacher (2017) also attempt to set out what students should be able to do at different ages. In their view, learners up to grade 4 are supported to find solutions to particular problem instances and to develop general strategies only on a very intuitive level. For older students (grades 7 to 9), it is not recommended to teach specialised concrete algorithms that do not work if the problem specification is changed just a little. In later work, Zhang et al. (2020) provide a CT skills progression in grades 1-9 in Sweden. To doing so, they activate and test some CT skills. For example, students in grades 1 to 3 use simple commands such as directions and steps, or repeat one command several times to compose more extended sequences. In grades 4 to 9, students use sequences to perform more complex tasks, such as initiating values to different variables, or calculate arithmetic.

5.6 How CT and related concepts are assessed

The introduction of Computation Thinking (CT) skills in compulsory education has fostered research and development of assessment methods and tools (e.g., Djambong et al., 2018; Tang et al., 2020; Tikva & Tambouris, 2021; Zhang et al., 2020).

While it is generally agreed that CT skills are more [easily / likely to be] developed as part of a separate Computer Science/Informatics subject, when it comes to classroom implementation there is a need to make abstract concepts concrete by embodying them in practice. Programming provides a laboratory for teaching and learning the basic CS/Informatics concepts underpinning CT skills. Particular attention should be paid to making explicit the design aspects of constructing an executable solution. Here, formative assessment can play an important role by proposing means for conceptualising a solution, not just for coding it. Similarly, artefact-based interviews can foster learners' reflections on the design and algorithmic aspects of their programming projects. In addition, automatic analysis of program code can provide the means for formative assessment of some CT skills (Grover, 2021). Recent research works also focus on identifying constructs, i.e., CT patterns[87] that act as abstraction, describing common object interactions in domains such as game design or simulations. CT patterns reveal the connections between programming concepts (e.g., loops, conditionals) and problem solving outside CS/Informatics domains (Repenning & Basawapatna, 2021).

To conduct effective formative assessment[88] of CT skills, teachers should be able to draw on a variety of tools, employing these in a manner that satisfies the constrains of timeliness and appropriateness. In some cases, formative assessment requires a quick return cycle to ensure timely feedback, and here automated tools (e.g., multiple choice, Parson problems, Dr Scratch, CT patterns) may play an important role. In other cases, reflective strategies would be more appropriate, thus calling for a different set of tools. These may include open ended questions, assignments or projects (e.g., showcases, reflection journals, rubrics). Therefore, formative assessment calls for a system of *assessment approaches*, i.e., combining different tools and instruments, each focused on different aspects of CT skills (Grover, 2021).

Turning to summative assessments,[89] international initiatives like the International Computer and Information Literacy Study (ICILS) and PISA 2022 Maths Framework highlight how the development of assessment items is strictly related to the definition of CT skills and how they are operationalised.

87. Computational Thinking Patterns (CTPs) are constructs that students initially learn in game design and can then apply to in creating simulations. Examples of CTPs include one agent tracking another agent, one agent absorbing another agent, and one agent creating another agent (Basawapatna et al., 2010).
88. Formative assessment (or assessment for learning) is not one specific practice, but rather an approach to teaching and learning. It can be best seen as a conceptual approach – a dynamic process that teachers adapt according to conditions and needs (Clark, 2010). Black and Wiliam (2010) define formative assessment as "all those activities undertaken by teachers — and by their students in assessing themselves — that provide information to be used as feedback to modify teaching and learning activities. Such assessment becomes formative assessment when the evidence is actually used to adapt the teaching to meet student needs." (Wiliam, 2011).
89. Summative assessment is the assessment of learners' achievement, usually at the end of a term, stage, course or programme, although not necessarily involving formal testing or examinations. Summative assessment is most commonly used for ranking, grading and/or promoting students, and for certification purposes (International Bureau of Education, UNESCO).

The conceptualisation of CT skills adopted in the International Computer and Information Literacy Study (Fraillon et al., 2019) focuses on problem solving to generate computer-based solutions.[90] From this conceptualisation, the ICILS CT framework derives two strands: conceptualising problems and operationalising solutions (Figure 12).

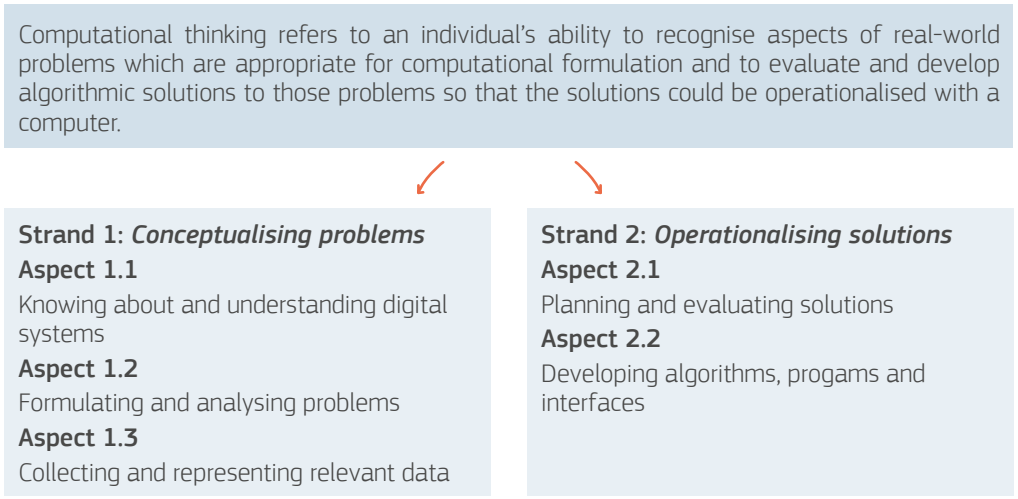


Figure 12. CT conceptualisation and strands in the ICILS CT 2018 framework
Source: Fraillon et al. 2019, p.28

The conceptualising-problems strand incorporates three aspects: (i) knowing about and understanding digital systems; (ii) formulating and analysing problems; and (iii) collecting and representing relevant data. The operationalising solutions strand encompasses two aspects: (a) planning and evaluating solutions; (b) developing algorithms, programs, and interfaces. This theoretical construct supports the formulation of test modules related to the two identified strands.

While ICILS CT 2018 assesses CT skills per se, PISA 2022 assesses CT within Maths, acknowledging CT as a key 21st century skill relevant for maths learning. For the development of the PISA assessment items, CT is defined as "thinking of solutions and expressing them in ways that can be automated by a machine involving algorithmic thinking, abstraction, problem decomposition, pattern recognition, evaluation of alternate solutions" (OECD, 2018)[91].

In the case-study interviews carried out for this study, primary teachers pointed out that the assessment of CT skills they perform is mostly formative. It is based on their observation of pupils while developing projects, solving problems, and learning by reflecting on mistakes. Group discussion conducted at the end of learning activities is used to prompt students to verbalise what they think, how they think, and how they arrive at solutions. The teachers also use quizzes (e.g., Bebras tasks), exercises, and surveys as tools for assessing CT skills.

As at primary level, the focus in lower secondary school is on formative assessment. Once again, a variety of approaches is used, including assessment scales and lists, e-portfolios, automated feedback, self-assessment, tasks or simulations, digital learning diaries, adaptive tasks, and peer assessment. For example, when doing project work, students are provided with a list of criteria that their solution should meet. For tasks like making a robot, the assessment starts with observing whether the robot performs as expected and required. At this level, summative assessment of CT skills takes on a more significant role and can include oral exams, written exams and/or computer-based exams, e-portfolios, student projects, online tests or applications. The main focus is on students' understanding of the programming task and proposed solutions.

In country cases that have final national exams at the end of lower secondary education, these are being updated to include CT skills. Provisions have already been made in France and Sweden to include CT skills in the final national exam of Maths & Technology at the end of ISCED 2. This step has already been enacted in France (Diplome du brevet – see Box 6). As interviewees in the lower secondary case studies made clear, the integration of programming in the final exam at the end of ISCED 2 is a strong indication of the importance attributed to this topic as part of the lower secondary curriculum.

90. It is worth noting that the ICILS CT Framework conceptualisation (i.e., problem-solving to generate computer-based solutions) is in line with that proposed in the present work, namely that CT skills develop through learning of basic CS/Informatics concepts related to *algorithms and programming*.
91. https://www.oecd.org/pisa/publications/pisa-2021-assessment-and-analytical-framework.htm

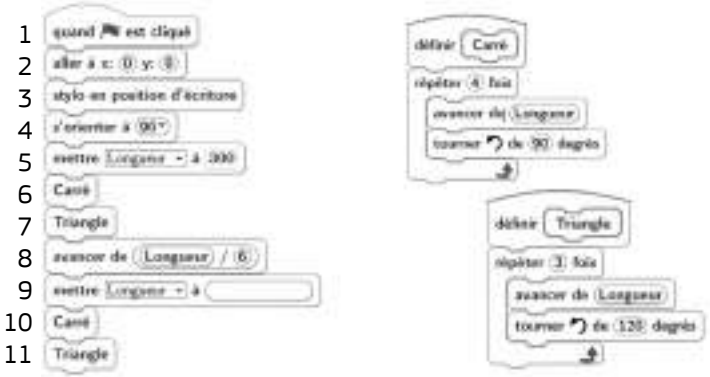
Box 6 Example of a test item from the French Diplôme National du Brevet for Mathematics (2018 session)

Source: Authors' translation into English

France's Ministry of Education has a policy of publishing previous exams as examples for preparing students for the Diplôme National du Brevet. The following exercise is taken from the text of the 2018 edition of the Maths exam (a written test). It comprises seven exercises for a total of 100 points. Exercise 6 deals with algorithms and programming: the English translation is shown below.

Exercise 6 (16 points)

The lengths are in pixels.
The expression "point in direction 90" (s'orienter à) means that you are facing to the right.
The following program is given:



1. The scale is 1 cm for 50 pixels.
   a. Show on your copy the figure obtained if the program is executed up to and including line 7.
   b. What are the coordinates of the pen after running line 8?
2. The complete program is executed, and the figure below is obtained which has a vertical axis of symmetry.

Copy and complete line 9 of the program to obtain this figure.

3.
   a. Among the following transformations (translation, homothety, rotation, axial symmetry), which one allows you to obtain the small square from the large square? Specify the reduction ratio.
   b. What is the ratio of the areas between the two squares drawn?

Where a mandatory final exam is not foreseen at the end of compulsory education, a more detailed set of criteria for summative assessment. For example, Finland recently revised the criteria for the final assessment of basic education in all subjects. This revision was introduced to provide a more homogenous and reliable basis for grading students' results at the end of lower secondary school. Criteria for assessing learning of algorithmic thinking and programming skills in grades 7-9 were specified, listing the requirements for assigning the mark levels 5, 7, 8 and 9 (see Box 7).

Box 7 Guidelines for final assessment of students' learning and skills in algorithmic thinking and programming for grades 7-9 in Finland

Source: Authors' translation of final assessment for grades 7-9 in Finland

Objectives of instruction	***O20*** – Guiding student development of algorithmic thinking and skills so they can apply mathematics and programming to solve problems.
Content areas related to the objectives	***C1*** "Thinking skills and methods: the students practise activities requiring logical thinking, such as discovering rules and dependencies and presenting them accurately. They consider and determine the number of possible alternatives. The students' reasoning and argumentation skills are strengthened. The students practise interpreting and producing mathematical notations. They familiarise themselves with the basics of providing proof. They practise determining the truth value of propositions. The students deepen their algorithmic thinking. The students programme while learning good programming practices. They use their own or ready-made computer programmes as a part of learning mathematics."
Teaching objectives derived from the learning objectives	Students will understand the principles of algorithmic thinking. They can read, annotate, interpret, test, design and implement small programs to solve mathematical problems.
Assessment targets in the subject	Algorithmic thinking and programming skills
Description of competences required for a 5 mark	The student identifies the steps of a simple algorithm and tests ready-made programs under supervision.
Description of competences required for a 7 mark	The student uses the conditional and repetition structure in programming, and test and interpret programs.
Description of competences required for a 8 mark	The student applies the principles of algorithmic thinking and program small programs.
Description of competences required for a 9 mark	The student uses programming to solve problems. The student will modify and develop a program.

Assessment tools

In the literature, tools for assessing Computation Thinking (CT) skills are mostly grouped into the following categories (Román-González et al., 2017a; Djambong et al., 2018):

1. CT summative tools – the aptitude-based Computational Thinking Test (Román-González et al., 2017b), the Test for Measuring Basic Programming Abilities (Mühling et al., 2015), the Commutative Assessment Test (Weintrop & Wilensky, 2015);
2. CT formative-iterative tools, which aim to provide students with immediate automatic feedback, helping them to improve their abilities. Examples in this category are Dr Scratch (Moreno-León & Robles, 2015) and the Computational Thinking pattern graph (Koh et al., 2010);
3. CT skill transfer tools – the tasks in international Bebras competitions (www.bebras.org), and the Computational Thinking Pattern Quiz (Izu et al., 2017; Basawapatna et al., 2011);
4. CT perception-attitude scales, i.e., the Computational Thinking Scales (CTS) developed by Korkmaz, Çakir, and Özden (2017);
5. CT vocabulary assessment, i.e., tools that can help measure students' verbal skills when doing coding tasks (Grover, 2011).

It is worth noting that "the information coming from each type of instrument has a different nature and all of them must be harmonised and triangulated to reach a complete CT assessment of the individual" (Román-González et al., 2019, p. 83). Application, validation, and discussion of the assessment tool called CTt (Computational Thinking test) developed by Román-González, Moreno-León, & Robles (2017a) was reported in several papers in this study's literature review (Guggemos, 2021; Román-González et al., 2017b). The CTt focuses on the following components: "sequences; loops; events; parallelism; conditionals; operators; data computational practices; problem-solving practices that occur in the process of programming; experimenting and iterating required task; testing and debugging; reusing and remixing; abstracting and modularizing" (Román-González et al., 2017a, p. 679).

The Bebras competition involves analyse and apply levels of the taxonomy proposed by Labusch & Eickelmann (2020, p. 3). Bebras tasks are used as another assessment tool for CT skills (Labusch & Eickelmann, 2020; Australian Computing Academy, 2019; Djambong et al., 2018). Dr Scratch and Scratch are tools that can be used for assessing CT skills (Australian Computing Academy, 2019; Fagerlund et al., 2020; Grover et al., 2019; Labusch & Eickelmann, 2020; Wei et al., 2021). Dr Scratch is an open-source CT assessment tool that automatically analyses the Scratch programming projects that students submit.

5.7 Technologies for teaching, learning & assessing Computational Thinking skills

As mentioned in Section 2.3, the nine in-depth case studies undertaken in this study covered the policy perspective (national statutory curriculum for compulsory education, policy guidelines and similar official support material) complemented by the view seen from schools (interview with one school leader and one teacher from each country). When it comes to technologies for teaching, learning and assessing Computational Thinking skills, the case studies help us to:

i. understand what is identified, recommended, prescribed, and possibly provided for CT-related use in those countries, and
ii. get a glimpse of what teachers and learners are actually adopting for use.

Looking at this aspect across the nine case studies, similarities emerge regarding the type of technologies identified, adopted and used for developing learners' CT skills – explicitly or implicitly – irrespective of the educational modality pursued (see Table 9).

Both screen-dependent and physical computing systems feature. The screen-dependent technologies mostly entail a programming dimension that may fall into one of two distinct categories: (i) block-based visual programming environments such as Scratch, CodeMonkey and StoryJumper, or (ii) textual programming languages like Logo and Python. On the other hand, physical computing systems cover systems for (programmable) computer control of physical robotic objects or regard kits (like Beebot and Arduino) for building controllable objects from scratch with modular components.

In most of the case studies, systems that are popular and widely were mentioned, some being open source and others commercially distributed. However, some "home-grown" systems like EMA and Salaby were also identified and discussed. It should be noted that this study's analysis of digital learning tools and technologies actually used in classrooms leaves aside the design and development of experimental systems.

The tools used fall into two major categories: *visual programming environments* and *robotics programmable kits*, as detailed in Table 8. Locally developed tools and environments are present.

	Primary	Lower secondary
Visual programming languages	CodeMonkey, StoryJumper, Xlogo	
	Scratch, Minecraft	
Textual programming languages	Imagine Logo	Python, Logo, Baltie
Physical computing	Beebot, Blue Bot, Pro-Bot, Lego (Wedo and Spike), Sphero	Mbot, Arduino, BeCreo
	Micro:bits, Makey Makey	
Locally developed tools & environments	EMA, Salaby	
	Ville/Eduten, Bebras task, Bebras task cards	

Table 9. Programming environments and physical computing tools that emerged from the nine case studies
Source: Authors' elaboration

5.8 Ensuring broader development of CT skills via gender balance, equity & inclusion

Making sure all learners in compulsory education have the opportunity to develop CT skills means acknowledging and tackling the tightly intertwined issues of gender balance, equity and inclusion. To address these matters effectively, it is imperative that they be taken seriously on board by all key stakeholders: the research community, policy makers, educational authorities and decision makers, school leaders, teachers, families, educational organisations and enterprises, and so on. Very generally speaking, the multiple perspectives gathered together in this study (representing a number of those same stakeholder groups) tend to see a gender-balanced, equitable and inclusive approach to CT skills development in compulsory education in terms of *access*. Here, there is general recognition that when CT is part of mandatory study of computing-related topics (Computer Science, Informatics, Maths, etc.), equal access for all students is more-or-less adequately ensured. However, the same does not necessarily apply to *elective study* of those topics, a situation that is far less common in compulsory education systems but nevertheless does occur to some degree, for instance in Croatia, where Informatics is compulsory in grades 5 and 6 and elective in grades 7 and 8).

From the research perspective, the study of Computational Thinking Educational Policy Initiatives (CTEPI) across the globe conducted by Hsu et al. (2019) notes a general trend to push for broader access and enhanced learner interest via compulsory CT education for all students. The authors argue that this is based on two premises: "CT is a foundational skill that all students should have to be digitally competent and active participants in a world where computing is pervasive and from a desire to motivate interest in CS and STEM, especially among girls and underrepresented minorities." (Webb et al., 2017, in Hsu et al., 2019, p. 268).

By the same token, Cateté et al. (2020) warn that a shortage of qualified teachers can be a barrier to providing all students with more equitable access to CS education. They stress that teachers' professional development should prepare them to teach CS to cohorts of students with diverse ethnicities, socioeconomic backgrounds and genders.

In the case studies, which embraced both policy and practice perspectives, it was generally considered that where CT is part of – or intrinsically linked to – compulsory studies, equal access for all students is suitably and adequately ensured. Indeed, in the Norwegian Case Study, ensuring equality of access through connection with purely compulsory studies was mentioned as a potential driver for incorporating informatics and programming at primary school level. The situation regarding computing-related subjects or topics made available as electives did not figure prominently: among the nine case studies conducted, the only one that included some elective study component (in lower secondary school) is Croatia. The Croatian interviewees asserted that, where Informatics is offered as an elective subject, girls are well represented. However, at the same time it was found that there might be a dividing line between more capable students and those who tend to struggle. A similar view was also echoed in the UK-England case study.

At the same time, however, there appears to be a general lack of system-level data about which students opt for elective CS/informatics subjects, and what comparative degree of success they achieve. So this is a matter that clearly needs to be addressed in order to get a firmer understanding of how gender/equity/inclusion issues impact on the quality of computing education.

Gender Balance

The research on CT skills development and gender in compulsory education often explores the relationship in terms of the different components that are recognised to comprise CT (see Section 3). Taking CT as a whole, no clearly unequivocal position seems to emerge: some studies find no significant gender imbalance (Atmatzidou, & Demetriadis, 2016; del Olmo-Muñoz et al., 2020; Tsai et al., 2020; Witherspoon et al., 2017; Wu & Su, 2021), while others find gender differences in particular approaches to development of CT skills, possibly related to the specific abilities activated by given tasks (Román-González et al., 2017b). Tsai et al. (2020) found that boys had a significantly higher disposition than girls in decomposition thinking, while Wu & Su (2021) reported primary school girls returning slightly higher scores for four CT dimensions (decomposition, pattern recognition, abstraction, algorithm design). Rijke et al. (2018, p. 85) found that at around ten years of age, girls begin to outperform boys in abstraction. This finding is consistent with a study by Israel-Fishelson et al. (2021), which found girls to be noticeably more advanced than boys in terms of both creative thinking and computational creativity (p. 1436). Research results reported by Guggemos (2021, p. 12) show that motivation impacts strongly on CT and gender differences. Sun et al. (2021, p. 355) found that primary school girls have a more positive attitude to STEM, with slightly higher CT skill development rates as well. They conclude that different gender-based developmental rates and phases may be a key determining factor here. Indeed, maintaining the interest of girls (and underrepresented minorities) in Computer Science and STEM is held to be a driver for treating Computation Thinking as a foundational skill for all (Hsu et al., 2019, p. 268). In support of the task dependency hypothesized by Román-González et al. (2017b), Kong et al. (2018) report boys being more interested in programming, and recommend that teachers should seek to foster girls' motivation towards this particular CT-related activity. Switzerland also runs several initiatives to attract girls to CS generally. Attempts by the Israeli Ministry of Education to introduce CS to girls in middle schools have, however, met with limited success, as about only a third of the students in these courses are girls.

Equity

When discussing ways to foster equitable and inclusive access to CT development, Huang and Looi (2020) highlight the large number of activities that have been designed and adapted to be run in unplugged mode. They cite these as evidence of the approach's flexibility and suitability for a wide range of learners, including students from low-income backgrounds and those who may have limited or no access to computers or Internet connection. The authors' literature review examines several studies into CS education equity; these focus on identity themes like self-perception, social recognition, authorship, purpose, status, and interests outside of school (Barron, 2006; Fields & Enyedy, 2013; Pinkard et al., 2017; Ryoo et al., 2013). They note that "unplugged activities appear in curricula that are specifically designed for girls, students of colour, students with special needs, and students in low-income communities, but there have been no studies that theorized the rationale for their inclusion" (Huang & Looi, 2020, p. 97). Accordingly, based on their investigation, they propose a model for better addressing the development of underrepresented student identities. And in their view, this new focus is a way to spark and sustain students' interest in CS generally.

Huang & Looi's model represents unplugged activities according to three different dimensions: content, pedagogy, and purpose of CS teaching. In the first case, they advocate expanding the scope of activities so that they cover content areas that really matter to underrepresented students, especially with regard to identity markers such as race, gender, socioeconomic status, and disability. These activities should build on [their] interests and involve practical real-world conditions and associated problems, e.g., shape of the room, physical obstacles, battery life. As to pedagogy, the authors suggest augmenting inquiry-based approaches with equity-conscious teaching (and teacher support) strategies, namely: helping teachers to become aware of any unconscious bias they may have about students' abilities; actively engaging underrepresented students; and using unplugged activities to create an initial impression of CS as accessible, social, and intellectually challenging. Finally, with regard to the purpose of fostering students' core CT skills, the authors suggest aligning teaching with students' personal vision of their life-purpose and the concerns of their communities.

Data from the survey conducted for this study with representatives from 30 European countries revealed that in seven countries (BE nl, CH, DK, EL, FR, IL, RU) a specify strategy exists to ensure gender balance and/or equity regarding Computational Thinking skills. In France, the IOTA project is being run in conjunction with the femmesnumerique undertaking[92], with particular attention on establishing non-gendered dynamics in CT-oriented learning activities.

In the U.S., the non-profit organisation Code.org was established in 2013 to bring CS to the core curriculum, and specifically to increase the participation of women and minorities (Fisher 2016). The K12CS Framework lists broadening participation in computer science as one of its guiding principles, noting that the structure and content of the framework reflect the need for diversity in computing and attention to issues of equity, including accessibility (K-12 Computer Science Framework 2016, p. 15). Other CS initiatives have focused on programs specifically targeting girls and underrepresented minorities.

Inclusion

Regarding the inclusion of all students in CT-related activities, the limited research available mainly focuses on providing suitable specialised instruction for learners with Special Education Needs (SEN) (e.g., Snodgrass et al 2016; Taylor et al 2017). For example, Ray and colleagues (2018) conducted an experimental study identifying a set of key pedagogical strategies to provide access to students with disabilities during CS/Informatics activities. This approach was grounded on Universal Design for Learning (UDL), which guides teachers firstly in defining goals and barriers to learning, and then in developing flexible learning environments, resources, activities, and assessments that meet the needs of a broad range of learners, including those with disabilities. The students with special educational needs involved in the study faced challenges related to behaviour (e.g., impulsivity, becoming easily frustrated by complex activities), and to academic difficulties (e.g., difficulty with reading, challenges with generalising information from one activity to another, limited memory). The authors have defined a frame to support students with special education needs during CS/Informatics activities, combining three main pedagogical strategies: (a) promoting peer collaboration among learners; (b) considering UDL principles when planning for student choice and access; and (c) using explicit instruction. Regarding the first of these, multiple collaboration activities were implemented in a continuum that stretched from being highly structured by the teacher (e.g., pair programming, rotating students' roles, teacher-created handouts with questions, buddy system for peer tutoring) to more student-driven collaboration (e.g., working in pairs without teacher instructions). This approach highlights that the application of UDL principles in designing CS/Informatics educational activities allows teachers to offer students choice and options in the type of activity and/or materials used to accommodate students' physical and sensory differences. This approach helped increase students' interest, and also provided them with strategies and guidelines for project planning (e.g., protocols and checklists for guiding students in multi-step problem solving, information sheets on Scratch blocks used in projects, etc.). Finally, the third combined approach entailed explicit instructions aimed at simplifying and clarifying CS/Informatics activities tailored to the needs of students and type of tasks. Explicit instructions mainly include teachers' step-by-step demonstrations (e.g., allowing students to see what they are being asked to do), then the whole class working through examples together, and then students themselves working through examples as the teacher monitors and provides support. As highlighted by the authors, this combination of multiple collaboration strategies helps teachers to increase access to CS/Informatics activities for all students, including those with special education needs.

92. https://femmesnumerique.fr

Young learners with special education needs can greatly benefit from using popular visual programming environments such as Scratch (e.g., Das et al, 2020; Paniagua & Instance, 2018; Pinto et al, 2016). Nevertheless, the need remains to boost the development of new digital tools, resources and learning activities, and, at a broader level, ensure policy initiatives make adequate provision for the accessibility and usability needs of all learners, including those with disabilities. Such efforts could at least start with the adoption of Universal Design for All principles to make the user interface of programming environments more accessible to young learners who, for example, are unable to use a mouse or who rely on a text-to-voice screen reader to perceive the results of programming they have created. To address this need, several initiatives such as AccessCSforALL, Bootstrap, and CSforAll are making efforts to develop resources that help teachers in the endeavour to fully integrate students with disabilities in Computer Science activities and ensure that all learners have the opportunity to develop core CT skills.

6

Teacher recruitment and professional development

6.1 Shortage of qualified teachers to teach CT

According to Eurostat[93], in 2019, there were 5.2 million teachers employed in primary, lower secondary and upper secondary education in the 28 EU countries. The integration of Computational Thinking (CT) skills in the curriculum at all educational levels is creating **demand for large-scale teacher training schemes, both pre-service and in-service**.

Policy-making reports and work from the research literature generally indicate the lack of available teachers with a suitable background and possessing adequate capabilities as the key inhibitor to successful integration of CT skills into compulsory school curricula (e.g., Caeli & Yadav, 2020; Grover & Pea, 2018; Li, 2020; Royal Society, 2017; Webb et al., 2017; Zhang et al., 2020). The United Kingdom, one of the pioneer countries in integrating CT skills in compulsory education, faces a severe and growing shortage of computing teachers (Royal Society, 2017). The lack of qualified teachers[94] is also a barrier to providing all K-12 students in the United States with more equitable access to Computer Science (CS) education (Cateté et al., 2020). A possible explanation for this situation is that, unlike for Mathematics or Science, the population of CS teachers is not considerable, with usually only one present in each school. This makes recruiting particularly difficult and sets the issue as a top priority for integrating CT skills in compulsory education (Li, 2020).

When respondents to the Computational Thinking Educational Policy Initiatives (CTEPI) survey (Hsu et al., 2019) were asked to identify the main challenges faced in integrating CT at the different education levels, the lack of adequately trained teachers was mentioned most (83 times) and across all education levels. Other factors referred to were issues connected with assessment of CT/programming skills (54 times), lack of teaching tools and resources (44 times), and competition with other curriculum priorities (42 times). However, this differentiation was less marked in iVET (lower and secondary levels): here, the lack of adequately trained teachers (21), assessment issues (16), the lack of tools and resources (14), and competition with other curriculum priorities (7) were cited.

The experts engaged in the online and validation workshops run for this study emphasised that teachers are critical players in successfully integrating CT/Informatics/programming at any school level. Teachers need to be adequately assisted in fostering and assessing students' CT skills. Nevertheless, teachers must be capable of maintaining students' motivation and scaffolding them in developing their CT skills. In this study's workshops, the experts agreed that teachers in many countries still think about their teaching subject in an isolated way, which makes integrating CT skills across different subjects difficult.

93. https://ec.europa.eu/eurostat/web/products-eurostat-news/-/edn-20211005-1
94. "Qualified teacher" refers to any teacher with the competence to teach CS – and help learners develop CT skills – in a sound and effective manner, irrespective of how that competence was acquired. It does not assume the teacher possessing official certification of that competence, or having followed a certified course of study to achieve it, although that may indeed be the case.

As to teachers' capability levels, Zhang et al. (2020) point out the **lack of programming and CT knowledge as a reason why teachers do not include CT perspectives in their practices**. Thus, there is a clear need to further improve teachers' limited knowledge of CT and build their capacity on fundamental CT concepts and pedagogies. In addition, to successfully teach CS, teachers also need to be ready and willing to experiment, understand and accept that they know less than some of their students, and not be afraid to fail and learn from students – which is a big challenge for many teachers.

In this study's systematic literature review, Tikva and Tambouris (2021) identify three basic requirements for teachers' CT capacity building:

- availability of models describing/mapping the domain knowledge for teachers to help them approach the development of students' CT skills;
- coverage in undergraduate Initial Teacher Education (ITE) courses, with pedagogy and educational technology modules that foster learning and teaching for CT skill development;
- professional development initiatives, including specially designed workshops and training courses that further teachers' knowledge and skills.

The first of these, **educational domain-knowledge models**, could derive from the conjunction of conceptual-level investigations, such as those reported in Section 5, with the curriculum integration positioning, approaches and strategies described above.

As to the promotion of **CT capacity building in ITE**, the Australian Computing Academy (2019) points out that where coding and CT are treated as a core subject area like literacy and numeracy, this reduces the need for professional development interventions down the track and fosters the integration of CT skills into the teaching of a range of subjects. In addition, Lamprou and Repenning (2018) report dramatic rises in Swiss teachers' confidence to teach coding and develop CT skills after a 14-week ITE course taken to prepare them for the introduction of these subjects as mandatory for student study.

6.2 Key factors for successful professional development of teachers

Hazzan et al. (2020) stress that if Computational Thinking (CT) is regarded as a skill or set of skills that everyone needs (as argued in Section 3), this includes teachers themselves. The authors also highlight the leading role Computer Science (CS) teachers play in bringing educational change in their schools. However, without adequate support in the form of structured and ongoing professional development programmes, many teachers experience significant difficulties in dealing with CT, considering it an unfamiliar school subject (Royal Society, 2017, p. 6; Tang et al., 2020). Many researchers (e.g., Kong et al., 2017; Rich et al., 2021; Waite et al., 2020) advocate professional development interventions that focus on acquiring concepts and practices typical of CS and CT.

As to methodology, Grover et al. (2019) mention role-playing, hands-on activities, and lesson plan modification as effective ways to support teachers' deeper conceptual understanding of introductory programming concepts. Balanskat et al. (2018) identify online courses (particularly Massive Open Online Courses - MOOCs for short), university courses and inspirational conferences, while workshops of various type and duration are widely proposed (e.g., Razak et al., 2021). Indeed, Bower et al. (2017) found evidence that teachers participating in training workshops improve their basic grasp of CT content, pedagogy, and technology in a relatively short time, and (significantly) benefited from heightened confidence in dealing with these areas.

Kong, Lai and Sun (2020) identified four factors for successful teacher professional development:

- sustained periods of learning rather than mere attendance at a one-off workshop event;
- active participation in professional development rather than passive reception of knowledge;
- connection with opportunities for school practice and reflection embedded in a community of professional peer support and exchange of practice;
- focus on pedagogical content knowledge (PCK) that emphasises the teachability of content.

Jocius et al. (2020) propose a "3C" professional development model: Code (Bootcamp), Connect (connecting disciplinary content and pedagogy to CT), and Create (development of CT-infused learning segments). This approach supports shifts in teacher understandings of the role of CT in teaching across the curriculum and boosts self-efficacy.

A review of professional development for CT in US primary schools (Sherwood et al., 2021) identified three distinct approaches: (a) single teacher leader-driven model (STEM teacher teaming up with a professional development provider to run training sessions); (b) scaffolded professional development model, where school leaders and professional development providers work together; and (c) intensive coaching model, where school-based coaches and school leaders work with professional development providers to develop a CT integrated curriculum and lesson activities.

Several authors advocate the need for teachers to build their conception of CT and related practices (Australian Computing Academy, 2019; Hazzan et al., 2020; Hromkovič et al., 2016; Hsu et al., 2018). The theoretical soundness of the ideas and activities developed can be fashioned and checked through specific professional development interventions and then shared with peers.

As Box 8 below illustrates, the nine European countries investigated in the Multiple-Case Studies are deploying and/or planning various efforts to integrate CT in compulsory education. Ministries, national centres, teacher training institutions, and grassroots efforts are addressing the challenge of recruiting and training teachers to integrate the development of CT skills into educational practices.

Box 8 Overview of teachers' professional development on CT skills in nine case studies

Croatia

During preparation for implementing the current Informatics curriculum in Croatia, in January 2018 the first virtual classroom for teachers was opened on the Moodle platform by the Mentoring team from the Ministry of Science and Education. This virtual classroom space was designed to support primary and secondary school teachers in introducing the current Informatics curriculum. In addition, teachers reflected on their learning and teaching by participating in various activities and sharing their ideas and experiences with others. In the 2018-2020 period, 128 virtual classrooms were run continuously for over 50,000 participants, almost all teachers from Croatian primary and secondary schools.

Finland

Recruitment of new teachers does not apply as CT is integrated into existing subjects. Therefore, the focus – and one of the key challenges – is on upskilling teachers. In Finland, continuous professional development is provided mainly by universities, universities of applied sciences, service-industry organisations and companies. The Ministry of Education, through the National Agency for Education, finances the development and management of courses through competitive grants. This approach guarantees a wide variety and spread of training initiatives and opportunities that respond to specific trainee needs. Finnish teachers are required to fulfil an annual quota of 12-18 hours of professional development training annually, but this is for all subject areas, not just programming. The fact that a substitute class teacher needs to be found to fill in for any teacher attending a training event poses a challenge.

France

In France, the Ministry of Education offered a two-day compulsory training course for ISCED 2 Maths and Technology teachers in 2015 around the outset of the curriculum reform. In addition, a set of resources with guidelines, lesson plans, etc. was developed. Training courses are run by regional branches of the Ministry (*Acadamie*) and universities and private sector organisations. Since CT is integrated within the existing subjects of Maths and Technology, there has been no need to recruit new teachers in response to this curricular innovation.

Lithuania

In-service teacher training activities have been carried out by universities, the National Agency for Education, municipalities, and schools. First and foremost, support aims at encouraging teachers and supporting their initiative to teach Informatics/CT skills. Improving teachers' digital competences also remains an important issue. Institutions organise training seminars according to the needs of teachers, invite lecturers or, at the request of teachers, provide professional development courses in informatics education.

Norway

No new teachers are recruited to teach coding in primary school because primary school teachers teach almost all subjects. The Directorate of Education and Training is funding local authorities to provide their teachers with various training opportunities. Regional knowledge centres offer courses and competence development in CT and programming. Three Norwegian universities made a school-based (not individual learner-focused) MOOC called "Programming and CT". This is part of school development projects and comprises lessons, videos, different testing, and projects to try out. Furthermore, school leaders are provided with grants to purchase resources/equipment for teaching CT/programming.

Poland

The Polish CMI project[95] is aimed at primary and secondary school teachers, university teachers, and other adults showing a predisposition and interest to work with gifted students. The project seeks to develop student interest and motivation to explore the field of algorithmics and programming. By the end of 2020, 367 teachers were trained and 428 CMI extra-curricular activities were run involving a total of 5,620 students. The project is implemented in all provinces in Poland. Starting in the 2021-2022 academic year, the Ministry of Education and Science in Poland will fully support the professional development of CS teachers provided by CS faculties at universities. Four universities have already opened these courses, which cater for trainee teachers in general, and for pre- and in-service CS. The main emphasis is on implementation of the new CS curriculum, the constructionist method of learning, and CT as a competence underpinning problem solving with/without computers.

95. https://cmi.edu.pl

Slovakia

In general, teachers have not been adequately prepared to teach Informatics and programming to primary school students, although they have studied it as part of their ITE. Slovakia has therefore introduced programming environments specially designed for primary school years, enriched with teaching materials, and supported by training courses designed to meet teachers' needs appropriately. In addition, qualified teacher educators provide methodological (pedagogical) training. As a result, nowadays, an increasing number of Slovakian primary school teachers are no longer afraid to teach programming.

Sweden

In Sweden, the National Education Agency has offered online training courses and conferences about programming that target ISCED 2 Maths and Technology teachers (e.g., about programming), as well for all teachers in the system. In addition, several courses are developed and implemented by universities. Swedish teachers must fulfil an annual quota of 104 hours of professional development: this is regulated by agreements with the trade unions, and teachers' participation in professional development initiatives is managed and planned by the school leader.

UK-England

In UK-England, the National Computing Centre (NCC) develops high-quality teaching resources and teacher training for Computing teachers at the primary and secondary levels. Over two years, NCC developed a complete program of study called "Teach the Computing Curriculum"[96], containing 500 hours of lesson plans, assessment exercises, practical exercises, student interaction, progression charts, concept charts, teacher guides, etc. Resources and guidance are mapped for each school year. All materials are under an open license, free for anyone to use. The NCC has started to focus on whole-school engagement involving school leaders as well, rather than individual teachers only. The interviewed school leader suggested that the best support for computing teachers is professionalised networks, maybe even at the European level.

The main recommendations emerging from the multiple-case studies are to:

- provide training opportunities on a large scale that are fit for purpose, exploiting synergies between Ministries of Education and other organisations like grassroots organisations;
- include basic CS concepts in pre-service and in-service teacher training, presenting innovative pedagogical approaches to CT skills development;
- encourage targeted pre-service and in-service teacher training on programming and how to handle the progression between grades in teaching programming according to age and using age-appropriate tools;
- constantly update teacher support materials and other resources because in the CS field these almost always fall out of date quickly;
- foster a culture of safe mistake making in and through professional development so that teachers dare to experiment and innovate;
- support the development of a network that fosters professional collaboration among teachers, encouraging them to share experiences and concrete examples;
- monitor and research teacher training to identify how much and what training enables teachers to deliver CS education that attains the goals described in CS curricula, involving teachers as research partners;
- provide sustained funding for quality training as this is a precondition for enabling teachers to deliver high-quality, inclusive computing education.

96. https://teachcomputing.org/curriculum

7

Conclusions

To conclude this report, this section proposes a set of recommendations for fostering effective integration of CT skills in compulsory education. These recommendations are primarily aimed at supporting policy makers, but are also of relevance for other educational stakeholders. As illustrated below in Figure 13, they cover four critical areas of policy action for integrating CT skills in compulsory education: consolidated understanding, comprehensive integration, systemic rollout, and support policy.

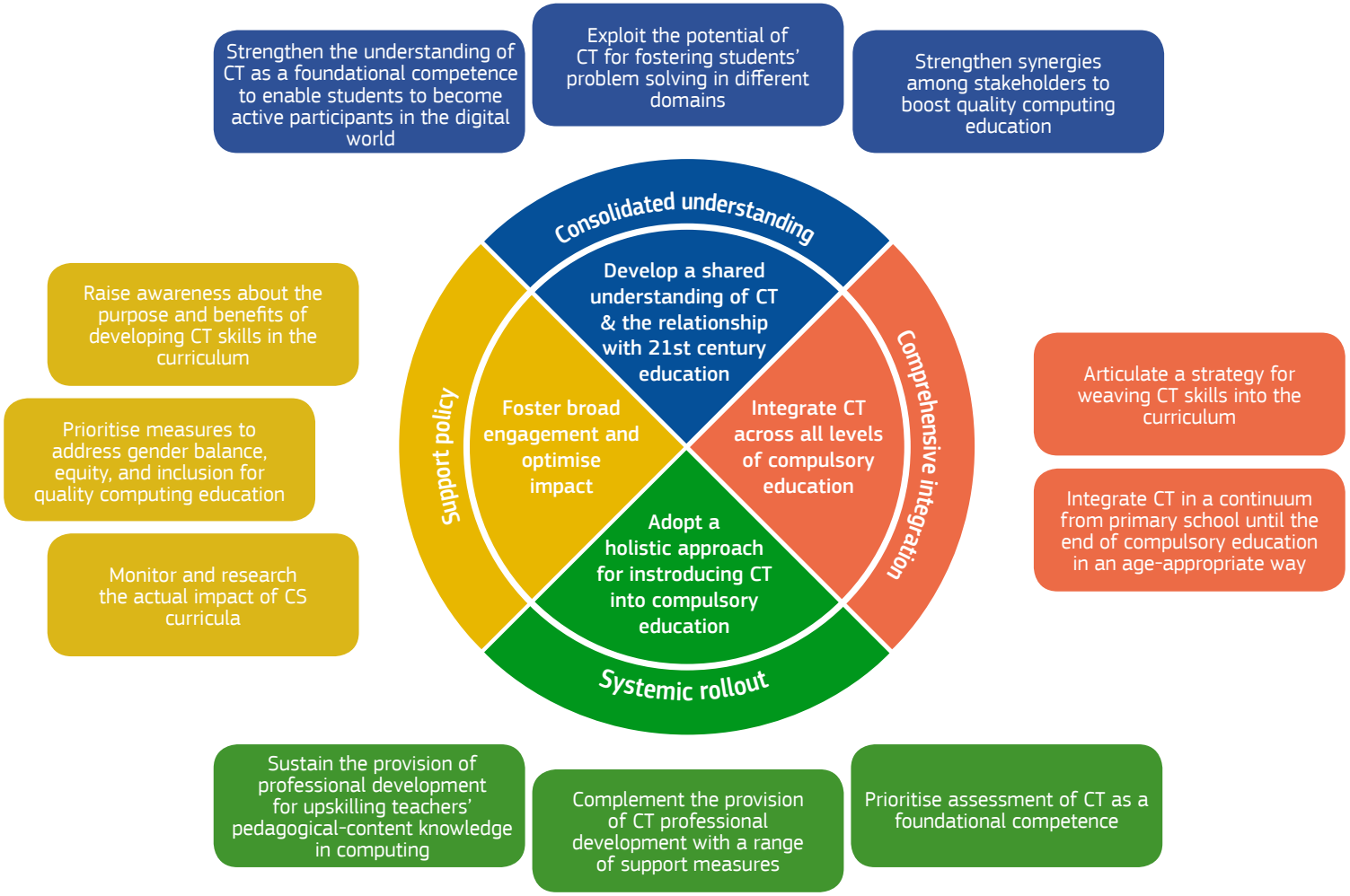


Figure 13. Policy recommendations for integrating CT skills into compulsory education
Source: Authors' elaboration

7.1 Policy recommendations

It is generally accepted that students will need digital skills to successfully navigate their professional and personal lives but also to function effectively as 21st century citizens. Digital skills are increasing becoming key, even to access basic public services such as administration and health but also to participate actively in formal and informal societal decision-making processes. So digital competence/literacy will be key, but it will not be sufficient. By learning basic Computer Science concepts and acquiring related Computational Thinking skills, students will be able to gain a clearer understanding of how the digital world around them works.

Eleven key recommendations have emerged from this study. These are presented and grouped according to the four above-mentioned areas of policy action for integrating CT skills in compulsory education. These four areas are the same ones considered in the earlier 2016 EC Computational Thinking report (Bocconi et al., 2016) as the outcome from this study suggests that they are still relevant today. At the end of this section, the eleven recommendations proposed here are compared with those that emerged from the 2016 report (see Table 10).

7.1.1 Consolidated understanding of Computational Thinking

Strengthen the understanding of CT as a foundational competence for students to become active participants in the digital world

Governments should foster curriculum implementation of CT by taking appropriate measures so that relevant educational stakeholders, school inspectors and evaluation agencies have a consolidated understanding of CT as a foundational competence. Digital skills are keenly required at a time when IT is pervading all aspects of society. It is essential to stress that these go beyond digital competence/digital literacy to include a basic scientific understanding of the digital revolution. Thus, the fundamental role of schools needs to be broadened, with literacy including the ability to solve problems in various fields through methods and tools derived from Computer Science (CT skills). The increasing complexity of the digital world is providing impetus for exploring the integration of more complex CS elements like machine learning and Artificial Intelligence in the final years of compulsory education. Extension in this direction necessarily means drawing on the latest research on how to address such topics in an effective and age-appropriate way.

Exploit the potential of CT for fostering students' problem solving in different domains

CT encompasses the thought processes entailed in formulating a problem for a computational solution. Rooted in Computer Science, CT skills can also be applied for solving problems in other domains, enabling students to create computational models of scientific phenomena, for example. For this to happen, teachers should provide effective scaffolding that helps learners make connections between computational approaches and essential characteristics of the application domain. Regardless of the curricula approach adopted for integrating CT skills (*cross-curricular theme, part of a separate subject, within other subjects*), prioritising the areas that can benefit most from creating such connections is crucial for boosting students' problem-solving skills in different domains.

Strengthen synergies among stakeholders to boost quality computing education

Building on the positive experience gained over recent years from coordination between compulsory education and grassroots initiatives, governments should set out a framework to strengthen synergies among stakeholders. This will contribute to the cultivation of interest in CT among parents and students, and support teachers in developing understanding of CT. More actions are needed to increase opportunities for sharing insights and knowledge among stakeholders, for reinforcing the capacity to reach out to students and parents, and for making policy-makers even more aware of CT as a set of key skills for understanding the digital world and pursuing future careers.

7.1.2 Comprehensive integration of CT skills in the curriculum

Articulate a strategy for weaving CT skills into the curriculum

The positioning of CT skills in the overall curriculum (per se or in an integrated manner) requires attention on several fronts. Governments should: make space in the curriculum for including foundational CS concepts (entailing algorithms and programming) to develop CT skills; provide clear guidelines on the amount of time teachers should devote to teaching basic CS contents; allocate resources for developing high-quality instructional material and examples of good pedagogical practices that go beyond programming; and make CT thinking and practice go hand-in-hand. When CT skills are positioned as a cross-curricular theme (i.e., addressed in all subjects taught), it is crucial to clarify the respective responsibilities of each subject teacher in this process. In addition, governments should provide sustained funding to ensure suitable digital equipment is available in all schools to support programming and educational robotics activities.

Integrate CT in a continuum from primary school until the end of compulsory education in an age-appropriate way

To guarantee overall consistency and progression in CT skills acquisition across grade levels, policy-makers should define a clear vision for the integration of CT in a continuum from the early years of primary until the end of compulsory education. Starting computing education at primary level enables students to progressively build foundational skills and gain an understanding of their potential application in other subjects. Through the years, teaching and learning activities should open the way not only to more sophisticated CT activities and tools (e.g., gradually progressing from visual programming languages to text-based languages) but also facilitate students' readiness to use CT skills in other domains.

7.1.3 Systemic rollout - adequate teacher support

Sustain the provision of professional development for upskilling teachers' pedagogical-content knowledge in computing

Governments should commit sustained investment and provision of high-quality professional development for teachers involving medium and long-term training interventions, enacted on a regular basis. Professional development opportunities need to focus on helping teachers gain the capacity to contextualise CT skills in their disciplines. Training on basic CS concepts – before moving further – should be made available, especially to teachers who are unfamiliar with computer sciences and/or as a way to involve less confident teachers (and schools). Coherent and systemic provision of teaching support should be established that offers methodological assistance, guidelines, good quality lesson plans, and appropriate scaffolding for reusing effective practices already tested in classrooms. Particular support should be provided to help teachers carry out both formative and summative assessment of students' CT skills. To ensure suitable teacher upskilling, sustained and substantial financial support should be made available to schools so they can release staff to take part in professional development initiatives. In particular, these should help teachers in handling the progression between grade levels, focusing on age-appropriate pedagogical approaches for teaching basic CS concepts. Guidelines should be provided that enable teachers to use suitable tools to that purpose, and to tailor their teaching and assessment to the needs and interests of (very able and less able) students. Furthermore, to encourage the application of CT as a set of problem-solving skills, teachers should be supported in spotlighting connections between topics in their specific subject and basic CS concepts.

Approaching the teacher upskilling challenge from a more long-term perspective, efforts should be devoted right now to including basic computing in pre-service education for compulsory school trainee-teachers.

Complement the provision of CT professional development with a range of support measures

Governments should sustain actions to complement CT professional development with the activation of support measures. Particular emphasis should be placed on fostering collaborative peer-support actions among teachers, such as networking and the sharing of concrete examples and experiences. Access should be provided to suitable, high-quality learning materials developed by different sources, like educational authorities, teachers, grassroots initiatives, and publishers. Furthermore, governments should form and sustain school hubs that connect schools for mutual support, raising and spreading quality in computing education. They should also ensure cross-cooperation with grassroots organisations that can provide additional training opportunities.

Prioritise assessment of CT as a foundational competence

Governments should make provisions for summative assessment of CT skills so as to monitor their ongoing establishment as a foundational component of compulsory education. Tasks assessing CT skills development should be integrated into the final exam at the end of lower secondary school to provide a clear indication of the importance attributed to computing in compulsory education. Teachers should be provided with effective guidelines and methods for formative assessment so they can monitor students' learning progress, and detect and deal with any emerging misconceptions. Detailed criteria for the assessment of CT skills should be defined, encompassing both students' understanding of basic CS concepts and their CT skills.

7.1.4 Policy support

Raise awareness about the purpose and benefits of developing CT skills in the curriculum

Governments should sustain awareness-raising initiatives that highlight the benefits of developing students' CT skills. Such initiatives should address all educational stakeholder groups (school leaders, school inspectors, teachers, students, parents, policymakers, as well as employers) and do so in a targeted way. Afterschool coding clubs, for example, have been successful in cultivating interest among both students and parents. Governments should work with industry and grassroots organisations to implement such impactful initiatives (e.g., EU Codeweek, Bebras, Computing at Schools, Informatics for all) and increase their outreach capacity.

Prioritise measures to address gender balance, equity and inclusion for quality computing education

Governments should define a clear strategy to ensure gender balance, equity and inclusion in computing education. Achieving gender balance is an issue that needs to be further investigated and addressed, for example by exploring effective ways to establish non-gendered dynamics in activities for CT skills development. To ensure equal access to CS education, low-cost computing equipment must be

made available inside and outside the classroom in order to guarantee that all students, including those from disadvantaged backgrounds, can participate in CS activities. Teachers should be supported in applying Universal Design for Learning principles to develop inclusive CS materials and activities that accommodate students' physical, cognitive and sensory differences.

Governments should pay particular attention to the gathering of system-level data to get a firmer understanding of how gender/equity/inclusion issues impact on quality computing education for all.

Monitor and research the actual impact of integrating CT skills in curricula

Governments should monitor whether and how schools are teaching basic Computer Science concepts to all students. For Computer Science to earn its place as an established subject in education, more evidence is needed to substantiate that the implementation of computing curricula actually reaches identified goals and, more broadly, how this contributes to fostering problem-solving skills. To that end, it is essential that CS concepts and related CT skills are part of the national curriculum and of assessment and inspection processes. More ongoing, systematic monitoring and evidence-based evaluation of curricular implementation is key, as are further research efforts. Particular attention should be paid to the impact of integrating CT skills *within other subjects* (e.g., Maths & Tech), focusing on the way those subjects are taught as a consequence, how students' learning in those subjects is impacted, as well as whether students have developed CT skills and the underlined basic CS concepts. Hence, policymakers and funders of education research should develop a long-term research agenda for computing education in schools.

CONSOLIDATED UNDERSTANDING

Establish a shared understanding of what CT is and how this is contextualised	Strengthen understanding of CT as foundational for helping students become active participants in the digital world
Clarify the overlaps and distinctions between CT and digital competence	Exploit the potential of CT for fostering students' problem solving in different domains
Encourage grassroots initiatives to enter the policy discussion on CT	Strengthen synergies among stakeholders to boost quality computing education

The recommendations proposed in the 2016 report to consolidate the understanding of CT suggested establishing a shared understanding of what CT is in context, and to clarify the overlaps and distinctions between CT and digital competence. The findings in the 2021 study, particularly the experience that emerged from the nine case studies, shed new light about those overlaps and distinctions, drawing new evidence from the ways CT has been integrated in the curriculum and implemented in real contexts. If stakeholders constructively take into account that evidence from policy and practice, it can potentially increase shared views about CT and offer operational options about ways to integrate it into the curriculum with the purpose of supporting students' digital competence/literacy and their problem-solving skills.

COMPREHENSIVE INTEGRATION

Articulate a vision for integrating CT in compulsory education, with clear goals	Articulate a strategy for weaving CT skills into the curriculum
Adopt a robust strategy for CT placement in the curriculum	Integrate CT in a continuum from primary until the end of compulsory education in an age-appropriate way
Include CT concepts and activities from early ages	

In the last five years, the discussion about the convergence between CT and digital competences (recommended in the 2016 report) opened up to the convergence between CT and the development of problem solving skills. This evolution is observed in research and reflected in policy initiatives at curriculum level (see above under Recommendation 2). Those developments represent effective progress that still needs to be pushed further as a way to define what a developmental view of CT teaching and learning could look like; which suitable assessment approaches could be adopted; and in what way the provision of effective pedagogical resources and programmes could be sustained. The recommendations proposed in the 2016 CompuThink report (Bocconi et al., 2016) suggested articulating a vision for integrating CT with clear goals and adoption of a robust strategy for CT integration in the curriculum. The findings in the 2021 study suggest further articulation of the vision by placing CT in a broader perspective, acknowledging the fact that most education systems integrating CT in their curriculum do so for developing students' problem-solving skills. The 2016 recommendations also suggested the inclusion of CT concepts and activities from early ages. This continues to be part of the 2021 recommendations, but is treated in a more systemic way, insisting on the need for consistency and scaffolding across all education levels and contexts.

Recommendations included in 2016 CompuThink report

Recommendations included in 2022 CompuThink report

SYSTEMIC ROLLOUT

Review and adapt innovative assessment methods	Prioritise assessment of CT as a foundational competence
Provide adequate support to teachers	Sustain provision of professional development for upskilling teachers' pedagogical-content knowledge in Computing
	Complement the provision of CT professional development with a range of support measures

The 2016 recommendation to provide adequate support to teachers is reiterated in the 2021 recommendations, but on a stronger, multiple-source evidence basis that draws from research, policy and practice. The analysis reveals that the demand for more training gives room to a demand for better training, tailored to each education level, accounting for more diversified contexts, and reflecting understanding of CT skills development (via learning of basic CS contents) that sustains students' digital competence and problem solving. The analysis suggests that researchers, policymakers and practitioners see the development of a collaborative ecosystem as invaluable in several respects, not least in bringing about more practice related/inspired approaches, and offering suitable responses to the multiple challenges generated by demand for increased training provision. More specific questions are on the table about how to deliver training on pedagogical-content knowledge in computing for teachers, how to ensure satisfactory participation and engagement in such initiatives, and what type of training opportunities are best suited to targeted teaching populations. At first sight, the 2021 recommendation about the assessment of CT skills might somehow appear still close to those proposed in the 2016 report, suggesting review and adaptation of innovative assessment methods. In addition to the thorny challenge of bringing innovation to the area of assessment in general, doing so for assessment of CT skills in particular is probably an even harder task. Indeed, it concerns a skill that is still new to compulsory education, and as such very much in need of clarification about the purpose and aim of its integration in the curriculum.

SUPPORT POLICY

Consolidate national and international exchanges on CT among policy makers, grassroots initiatives, research centres and other stakeholders	Raise awareness about the purpose and benefits of developing CT skills in the curriculum
Adopt a robust strategy for CT placement in the curriculum	Prioritise measures to address gender balance, equity and inclusion for quality computing education
Include CT concepts and activities from early ages	Monitor and research integration of CT skills in curricula

The recommendations in the 2016 report advocated (a) consolidation of national and international exchanges on CT among policy makers, grassroots initiatives, research centres and other stakeholders, (b) informing all relevant stakeholders, and (c) prioritising the follow-up to strengthen impact. Unsurprisingly, the two recommendations about consolidating exchanges and informing all stakeholders remain applicable in 2021. These actions are relevant in almost all phases and contexts, especially in a world evolving at high speed, in which policy action is no longer solely in the hands of policy makers, but also necessarily involves other stakeholders with significant roles to play in change. Interestingly, as the integration of CT skills in curricula has progressed, examples of ways to put such recommendations into practice now exist and are provided. The 2021 study presents significant advancement on the 2016 recommendation concerning follow-up to strengthen impact. Progress in the conceptualisation of CT as a thinking process to be taught in compulsory education (see Recommendations 1, 2 and 4) now provides clearer directions and focus for appropriate monitoring, taking into account the intrinsic complexity involved.

Table 10. Comparison of policy recommendations with those from the 2016 CompuThink report
Source: Authors' elaboration

References

Aho, A. V. (2012). Computation and Computational Thinking. *The Computer Journal, 55*(7), 832–835. https://doi.org/10.1093/comjnl/bxs074

Arfé, B., Vardanega, T., & Ronconi, L. (2020). The effects of coding on children's planning and inhibition skills. *Computers & Education, 148*. https://doi.org/10.1016/j.compedu.2020.103807

Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems, 75*, 661–670. https://doi.org/10.1016/j.robot.2015.10.008

Australian Computing Academy. (2019). Coding and Computational Thinking—What is the Evidence? Retrieved from https://education.nsw.gov.au/content/dam/main-education/teaching-and-learning/education-for-a-changing-world/media/documents/Coding-and-Computational-Report_A.pdf

Balanskat, A., Engelhardt, K., & Licht, A. H. (2018). Strategies to include computational thinking in school curricula in Norway and Sweden. European *Schoolnet's 2018 Study Visit*. European Schoolnet. Retrieved from https://docplayer.net/99728690-Strategies-to-include-computational-thinking-in-school-curricula-in-norway-and-sweden.html

Barron, B. (2006). Interest and Self-Sustained Learning as Catalysts of Development: A Learning Ecology Perspective. *Human Development, 49*(4), 193–224. https://doi.org/10.1159/000094368

Basawapatna, A. R., Koh, K. H., Repenning, A. (2010, June). Using scalable game design to teach computer science from middle school to graduate school. In *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education* (ITiCSE '10), (pp. 224–228). Association for Computing Machinery. https://doi.org/10.1145/1822090.1822154

Basawapatna, A., Koh, K. H., Repenning, A., Webb, D. C., & Marshall, K. S. (2011). Recognizing computational thinking patterns. In *Proceedings of the 42nd ACM technical symposium on Computer science education*, 245–250. Association for Computing Machinery. https://doi.org/10.1145/1953163.1953241

Basu, S., Biswas, G., & Kinnebrew, J. S. (2017). Learner modeling for adaptive scaffolding in a Computational Thinking-based science learning environment. *User Modeling and User-Adapted Interaction, 27*(1), 5-53. https://doi.org/10.1007/s11257-017-9187-0

Black, P., & Wiliam, D. (2010). Inside the Black Box: Raising Standards through Classroom Assessment. *Phi Delta Kappan, 92*(1), 81–90. https://doi.org/10.1177/003172171009200119

Blaskó, Z., da Costa, P. & Schnepf, S., (2021). Learning Loss and Educational Inequalities in Europe: Mapping the Potential Consequences of the COVID-19 Crisis. *IZA Discussion Papers Series*, Article 14298. https://doi.org/10.2139/ssrn.3833230

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing Computational Thinking in Compulsory Education - Implications for policy and practice*. Publications Office of the European Commission. https://doi.org/10.2791/792158

Bower, M., Wood, L., Lai, J., Howe, C., & Lister, R. (2017). Improving the Computational Thinking Pedagogical Capabilities of School Teachers. *Australian Journal of Teacher Education, 42*(3), 53–72. https://doi.org/10.14221/ajte.2017v42n3.4

Cachia, R., Velicu, A., Chaudron, S., Di Gioia, R. & Vuorikari, R. (2021). *Emergency remote schooling during COVID-19: a closer look at European families*. Publications Office of the European Union. https://doi.org/10.2760/613798

Caeli, E. N., & Yadav, A. (2020). Unplugged Approaches to Computational Thinking: A Historical Perspective. *TechTrends, 64*(1), 29–36. https://doi.org/10.1007/s11528-019-00410-5

Campe, S., Denner, J., Green, E., & Torres, D. (2020). Pair programming in middle school: variations in interactions and behaviors. *Computer Science Education, 30*(1), 22–46. https://doi.org/10.1080/08993408.2019.1648119

Cateté, V., Alvarez, L., Isvik, A., Milliken, A., Hill, M., & Barnes, T. (2020). Aligning Theory and Practice in Teacher Professional Development for Computer Science. *ACM Proceedings. 20th Koli Calling Conference on Computing Education Research*. https://doi.org/10.1145/3428029.3428560

Chevallard, Y. (1992). A Theoretical Approach to Curricula. *Journal Für Mathematik-Didaktik, 13*(2–3), 215–230. https://doi.org/10.1007/BF03338779

Ching, Y. H., Hsu, Y. C., & Baldwin, S. (2018). Developing Computational Thinking with Educational Technologies for Young Learners. *TechTrends, 62*(6), 563-573. https://doi.org/10.1007/s11528-018-0292-7

Clark, I. (2010). Formative Assessment: "There is Nothing so Practical as a Good Theory". *Australian Journal of Education, 54*(3), 341–352. https://doi.org/10.1177/000494411005400308

Corradini, I., Lodi, M., & Nardelli, E. (2017). Conceptions and Misconceptions about Computational Thinking among Italian Primary School Teachers. *ICER – Proceedings of ACM Conference. International Computing Education Research* (pp. 136–144). https://doi.org/10.1145/3105726.3106194

Crick, T. (2017). *Computing education: An overview of research in the field. Royal Society*. Retrieved from https://royalsociety.org/-/media/policy/projects/computing-education/literature-review-overview-research-field.pdf

Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T., Selby, C., & Woollard, J. (2015). *Computational thinking A guide for teachers. Computing at School*. Retrieved from https://www.semanticscholar.org/paper/Computational-thinking-a-guide-for-teachers-Csizmadia-Curzon/b8741512fec1821197c2f54e0bfadbdfef2d5302

Curzon, P., Bell, T., Waite, J., & Dorling, M. (2019). Computational Thinking. In A. V. Robins & S. A. Fincher (Eds.), *The Cambridge Handbook of Computing Education Research* (pp. 513–546). https://doi.org/10.1017/9781108654555.018

Das, M., Marghitu, D.B., Jamshidi, F., Mandala, M., & Howard, A.M. (2020). Accessible Computer Science for K-12 Students with Hearing Impairments. *ArXiv*, https://arxiv.org/abs/2007.08476

del Olmo-Muñoz, J., Cózar-Gutiérrez, R., & González-Calero, J. A. (2020). Computational thinking through unplugged activities in early years of Primary Education. *Computers & Education, 150*. https://doi.org/10.1016/j.compedu.2020.103832

Denning, P. J., & Tedre, M. (2021). Computational Thinking: A disciplinary perspective. *Informatics in Education, 20*(3), 361–390. https://doi.org/10.15388/infedu.2021.21

Djambong, T., Freiman, V., Gauvin, S., Paquet, M., & Chiasson, M. (2018). Measurement of computational thinking in K-12 education: The need for innovative practices. In D. Sampson, D. Ifenthaler, J. Spector, P. Isaías (Eds.), *Digital technologies: Sustainable innovations for improving teaching and learning* (pp. 193–222). Springer. https://doi.org/10.1007/978-3-319-73417-0_12

Eickelmann, B., Labusch, A., & Vennemann, M. (2019). Computational Thinking and Problem-Solving in the Context of IEA-ICILS 2018. In D. Passey, R. Bottino, C. Lewin, & E. Sanchez (Eds.), *IFIP TC 3 Open Conference on Computers in Education, OCCE 2018 Proceedings* (pp. 14–23). https://doi.org/10.1007/978-3-030-23513-0_2

European Commission. (2020). *Digital Education Action Plan 2021-2027: Resetting education and training for the digital age*. Retrieved from https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52020DC0624

European Commission/EACEA/Eurydice. (2020). *The Structure of the European Education Systems 2020/21 Schematic Diagrams*. Publications Office of the European Union. https://doi.org/10.2797/39049

European Commission/EACEA/Eurydice. (2019). *Digital Education at School in Europe*. Publications Office of the European Union. https://doi.org/10.2797/66552

Fagerlund, J., Häkkinen, P., Vesisenaho, M., & Viiri, J. (2020). Assessing 4th Grade Students' Computational Thinking through Scratch Programming Projects. *Informatics in Education, 19*(4), 611–640. https://doi.org/10.15388/INFEDU.2020.27

Fessakis, G., & Prantsoudi, S. (2019). Computer Science Teachers' Perceptions, Beliefs and Attitudes on Computational Thinking in Greece. *Informatics in Education, 18*(2), 227–258. https://doi.org/10.15388/infedu.2019.11

Fields, D., & Enyedy, N. (2013). Picking up the mantle of "expert": Assigned roles, assertion of identity, and peer recognition within a programming class. *Mind, Culture, and Activity, 20*(2), 113–131. https://doi.org/10.1080/10749039.2012.691199

Fisher, L. M. (2016). A decade of ACM efforts contributes to computer science for all. *Communications of the ACM, 59*(4), 25–27. http://doi.org/10.1145/2892740

Forlizzi, L., Lodi, M., Lonati, V., Mirolo, C., Monga, M., Montresor, A., Morpurgo, A., & Nardelli, E. (2018). A core informatics curriculum for Italian compulsory education. In S.N. Pozdniakov & V. Dagiene (Eds.), *Lecture Notes in Computer Science, 11169* (pp. 141–153). Springer. https://doi.org/10.1007/978-3-030-02750-6_11

Fowler, B. & Vegas, E. (2021). *How England implemented its computer science education program*. Center for Universal Education at Brookings. Retrieved from https://www.brookings.edu/research/how-england-implemented-its-computer-science-education-program

Fraillon, J., Ainley, J., Schulz, W., Duckworth, D., & Friedman, T. (2019). Computational thinking framework. In I*EA International Computer and Information Literacy Study 2018 Assessment Framework* (pp. 25–31). Springer. https://doi.org/10.1007/978-3-030-19389-8_3

Grover, S. (2021). Toward A Framework for Formative Assessment of Conceptual Learning in K-12 Computer Science Classrooms. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education (SIGCSE '21)* (pp. 31–37). Association for Computing Machinery. https://doi.org/10.1145/3408877.3432460

Grover, S. (2011). Robotics and engineering for middle and high school students to develop computational thinking. *Paper presented at the annual meeting of the American Educational Research Association, New Orleans*. Retrieved from https://www.shuchigrover.com/wp-content/uploads/2021/03/AERA2011-Shuchi-Grover-Robotics-and-Engineering-for-Middle-and-High-School-Students-to-Develop-Computational-Thinking.pdf

Grover, S. & Floyd, S. (2020). Question and Inquiry. In S. Grover (Ed.), *Computer Science in K-12: An A-To-Z Handbook on Teaching Programming* (pp. 180-188). Edfinity.

Grover, S., & Pea, R. (2018). Computational Thinking: A competency whose time has come. In S. Sentance, E. Barendsen, & S. Carsten (Eds.), *Computer Science Education: Perspectives on Teaching and Learning in School* (pp. 19–38). Bloomsbury Publishing.

Grover, S., & Pea, R. (2013). Computational Thinking in K-12: A Review of the State of the Field. *Educational Researcher, 42*(1), 38-43. https://doi.org/10.3102/0013189X12463051

Grover, S., Jackiw, N., & Lundh, P. (2019). Concepts before coding: Non-programming interactives to advance learning of introductory programming concepts in middle school. *Computer Science Education, 29* (2–3), 106–135. https://doi.org/10.1080/08993408.2019.1568955

Guggemos, J. (2021). On the predictors of computational thinking and its growth at the high-school level. *Computers & Education, 161*. https://doi.org/10.1016/j.compedu.2020.104060

Hazzan, O., Ragonis, N., Lapidot, T., & Rosenberg-Kima, R. (2020). Computational

Thinking. In O. Hazzan, N. Ragonis, & T. Lapidot, *Guide to Teaching Computer Science* (pp. 57–74). Springer. https://doi.org/10.1007/978-3-030-39360-1_4

Hromkovič, J., & Lacher, R. (2017). The Computer Science Way of Thinking in Human History and Consequences for the Design of Computer Science Curricula. In A. Hellas & V. Dagiene (Eds.), *Lecture Notes in Computer Science, 10696* (pp. 3–11). Springer. https://doi.org/10.1007/978-3-319-71483-7_1

Hromkovič, J., Kohn, T., Komm, D., & Serafini, G. (2016). Combining the Power of Python with the Simplicity of Logo for a Sustainable Computer Science Education. In *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives* (pp. 155–166). Springer. https://doi.org/10.1007/978-3-319-46747-4_13

Hsu, T.-C., Chang, S.-C., & Hung, Y.-T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education, 126*, 296–310. https://doi.org/10.1016/j.compedu.2018.07.004

Hsu, Y.-C., Irie, N. R., & Ching, Y.-H. (2019). Computational Thinking Educational Policy Initiatives (CTEPI) Across the Globe. *TechTrends*. https://doi.org/10.1007/s11528-019-00384-4

Huang, W., & Looi, C.-K. (2020). A critical review of literature on "unplugged" pedagogies in K-12 computer science and computational thinking education. *Computer Science Education*, 1–29. https://doi.org/10.1080/08993408.2020.1789411

Informatics Europe & ACM Europe. (2013). Informatics education: Europe cannot afford to miss the boat. Retrieved from https://www.informaticsforall.org/the-informatics-boat-reports/

Israel-Fishelson, R., & Hershkovitz, A. (2020). Persistence in a Game-Based Learning Environment: The Case of Elementary School Students Learning Computational Thinking. *Journal of Educational Computing Research, 58*(5), 891–918. https://doi.org/10.1177/0735633119887187

Israel-Fishelson, R., Hershkovitz, A., Eguíluz, A., Garaizar, P., & Guenaga, M. (2021). The Associations Between Computational Thinking and Creativity: The Role of Personal Characteristics. *Journal of Educational Computing Research, 58*(8), 1415–1447. https://doi.org/10.1177/0735633120940954

Izu, C., Mirolo, C., Settle, A., Mannila, L., & Stupuriene, G. (2017). Exploring Bebras Tasks Content and Performance: A Multinational Study. *Informatics in Education, 16*(1), 39–59. https://doi.org/10.15388/infedu.2017.03

Jocius, R., Joshi, D., Dong, Y., Robinson, R., Catete, V., Barnes, T., Albert, J., Andrews, A., & Lytl, N. (2020). Code, Connect, Create: The 3C Professional Development Model to Support Computational Thinking Infusion. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (pp. 971–977). https://doi.org/10.1145/3328778.3366797

K–12 Computer Science Framework. (2016). Retrieved from https://k12cs.org/wp-content/uploads/2016/09/K%E2%80%9312-Computer-Science-Framework.pdf

Kahn, K., & Winters, N. (2021). Constructionism and AI: A history and possible futures. *British Journal of Educational Technology*, 52, 1130–1142. https://doi.org/10.1111/bjet.13088

Kale, U., Akcaoglu, M., Cullen, T., Goh, D., Devine, L., Calvert, N., & Grise, K. (2018). Computational What? Relating Computational Thinking to Teaching. *TechTrends, 62*(6), 574–584. https://doi.org/10.1007/s11528-018-0290-9

Knuth, D. E. (1985). Algorithmic Thinking and Mathematical Thinking. *The American Mathematical Monthly, 92*(3), 170-181. https://doi.org/10.1080/00029890.1985.11971572

Koh, K. H., Basawapatna, A., Bennett, V., & Repenning, A. (2010, September). Towards the Automatic Recognition of Computational Thinking for Adaptive Visual Language Learning. In *IEEE Symposium on Visual Languages and Human-Centric Computing* (pp. 59–66). IEEE. https://doi.org/10.1109/VLHCC.2010.17

Komm, D., Hauser, U., Matter, B., Staub, J., & Trachsler, N. (2020). Computational Thinking in Small Packages. In K. Kori & M. Laanpere (Eds.), *Lecture Notes in Computer Science, 12518* (pp. 170–181). Springer. https://doi.org/10.1007/978-3-030-63212-0_14

Kong, S.-C., Chiu, M. M., & Lai, M. (2018). A study of primary school students' interest, collaboration attitude, and programming empowerment in computational thinking education. *Computers & Education, 127*, 178–189. https://doi.org/10.1016/j.compedu.2018.08.026

Kong, S.-C., Lai, M., Sheldon, J., & Tissenbaum, M. (2017). The design and evaluation of a teacher development programme in computational thinking education. In S.-C. Kong, J. Sheldon, & R. K.-Y. Li (Eds.,), *Proceedings of International Conference on Computational Thinking Education, CTE 2017* (pp. 77–80). The Education University of Hong Kong. Retrieved from https://repository.eduhk.hk/en/publications/the-design-and-evaluation-of-a-teacher-development-programme-in-c-7

Kong, S.-C., Lai, M., & Sun, D. (2020). Teacher development in computational thinking: Design and learning outcomes of programming concepts, practices and pedagogy. *Computers & Education, 151*. https://doi.org/10.1016/j.compedu.2020.103872

Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). *Computers in Human Behavior, 72*, 558–569. https://doi.org/10.1016/j.chb.2017.01.005

Kralj, L. (2016). New Informatics curriculum – Croatian tradition with world trends. In 2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO) (pp. 760-763). https://doi.org/10.1109/MIPRO.2016.7522242

Labusch, A., & Eickelmann, B. (2020). Computational Thinking Competences in Countries from Three Different Continents in the Mirror of Students' Characteristics and School Learning. In S. C. Kong, H. U. Hoppe, T. C. Hsu, R. H. Huang, B. C. Kuo, K. Y. Li, C. K. Looi, M. Milrad, J. L. Shih, K. F. Sin, K. S. Song, M. Specht, F. Sullivan, & J. Vahrenhold (Eds.), *Proceedings of International Conference on Computational Thinking Education 2020* (pp. 2–7). The Education University of Hong Kong. Retrieved from https://www.eduhk.hk/cte2020/doc/CTE2020%20Proceedings.pdf

Lamprou, A., & Repenning, A. (2018). Teaching how to teach computational thinking. In P. Andreou, M. Armoni, J.C. Read, & I. Polycarpou (Eds.,), *Annual Conference on Innovative and Technology In Computer Science ITiCSE* (pp. 69–74). Association for Computing Machinery. https://doi.org/10.1145/3197091.3197120

Li, Q. (2020). Computational thinking and teacher education: An expert interview study. *Human Behavior and Emerging Technologies*, 1–15. https://doi.org/10.1002/hbe2.224

Lee, I. (2011). Assessing Youth's Computational Thinking in the context of Modeling & Simulation. *AERA Conference Proceedings 2011*. Retrieved from the AERA Online Paper Repository http://www.aera.net/repository

Martinand, J.-L. (2014). Point de vue V – Didactique des sciences et techniques, didactique du curriculum. *Éducation et didactique, 8*(1), 65–76. https://doi.org/10.4000/educationdidactique.1886

Metcalf, S. J., Reilly, J. M., Jeon, S., Wang, A., Pyers, A., Brennan, K., & Dede, C. (2021). Assessing computational thinking through the lenses of functionality and computational fluency. *Computer Science Education*. https://doi.org/10.1080/08993408.2020.1866932

Moher, D., Liberati, A., Tetzlaff, J., Altman, D. G., & PRISMA Group. (2009). Preferred Reporting Items for Systematic Reviews and Meta-Analyses: The PRISMA Statement. *PLoS Medicine, 6*(7), 1–6. https://doi.org/10.1371/journal.pmed.1000097

Moreno-León, J., & Robles, G. (2015). Dr. Scratch: A Web Tool to Automatically Evaluate Scratch Projects. In *Proceedings of the Workshop in Primary and Secondary Computing Education* (pp. 132–133). https://doi.org/10.1145/2818314.2818338

Mühling, A., Ruf, A., & Hubwieser, P. (2015). Design and First Results of a Psychometric Test for Measuring Basic Programming Abilities. In *Proceedings of the Workshop in Primary and Secondary Computing Education* (pp. 2–10). https://doi.org/10.1145/2818314.2818320

OECD (2021). *Teachers and Leaders in Vocational Education and Training*, OECD Reviews of Vocational Education and Training. OECD Publishing. https://doi.org/10.1787/59d4fbb1-en

Paniagua, A. and D. Istance (2018). Teachers as Designers of Learning Environments: The Importance of Innovative Pedagogies, Educational Research and Innovation, OECD Publishing. https://doi.org/10.1787/9789264085374-en

Passey, D. (2017). Computer science (CS) in the compulsory education curriculum: Implications for future research. *Education and Information Technologies, 22*(2), 421–443. https://doi.org/10.1007/s10639-016-9475-z

Pinkard, N., Erete, S., Martin, C. K., & McKinney de Royston, M. (2017). Digital Youth Divas: Exploring Narrative-Driven Curriculum to Spark Middle School Girls' Interest in Computational Activities. *Journal of the Learning Sciences, 26*(3), 477–516. https://doi.org/10.1080/10508406.2017.1307199

Pinto, A., Gonçalo Oliveira, H., & Oliveira Alves, A. (2016). Comparing the performance of different NLP toolkits in formal and social media text. In *5th Symposium on Languages, Applications and Technologies (SLATE'16)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (pp. 3:1-3:16). https://doi.org/10.4230/OASIcs.SLATE.2016.3

Ray, M. J., Israel, M., Lee, C. E., & Do, V. (2018). A cross-Case Analysis of Instructional Strategies to Support Participation of K-8 Students with Disabilities in CS for All. SIGCSE – *Proceedings of ACM Technical Symposium on Computer Science Education* (pp. 900–905). https://doi.org/10.1145/3159450.3159482

Razak, S., Khan, S., Hussein, N., Alshikhabobakr, H., Gedawy, H., & Yousaf, A. W. (2021). Integrating Computer Science and ICT Concepts in a Cohesive Curriculum for Middle School-An Experience Report. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (pp. 966–972). https://doi.org/10.1145/3408877.3432528

Repenning, A., & Basawapatna, A. (2021). Explicative programming. *Communications of the ACM, 64*(11), 30–33. https://doi.org/10.1145/3486642

Resnick, M. (2017). Lifelong Kindergarten: *Cultivating Creativity through Projects, Passion, Peers, and Play*. The MIT Press. https://doi.org/10.7551/mitpress/11017.001.0001

Rich, P. J., Mason, S. L., & O'Leary, J. (2021). Measuring the Effect of Continuous Professional Development on Elementary Teachers' Self-Efficacy to Teach Coding and Computational Thinking. *Computers & Education, 168*. https://doi.org/10.1016/j.compedu.2021.104196

Rijke, W. J., Bollen, L., Eysink, T. H. S., & Tolboom, J. L. J. (2018). Computational Thinking in Primary School: An examination of Abstraction and Decomposition in Different Age Groups. *Informatics in Education, 17*(1), 77–92. https://doi.org/10.15388/infedu.2018.05

Román-González, M., Moreno-León, J., & Robles, G. (2017a). Complementary Tools for Computational Thinking Assessment. In S. C Kong, J Sheldon, and K. Y Li (Eds.), *Proceedings of International Conference on Computational Thinking Education - CTE 2017* (pp. 154–159). The Education University of Hong Kong. Retrieved from https://www.researchgate.net/publication/318469859_Complementary_Tools_for_Computational_Thinking_Assessment

Román-González, M., Pérez-González, J.-C., & Jiménez-Fernández, C. (2017b). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior, 72*, 678–691. https://doi.org/10.1016/j.chb.2016.08.047

Royal Society. (2017). After the reboot: Computing education in UK schools. *Policy Report*. Retrieved from https://royalsociety.org/~/media/policy/projects/

computing-education/computing-education-report.pdf

Ryoo, J. J., Margolis, J., Lee, C. H., Sandoval, C. D., & Goode, J. (2013). Democratizing computer science knowledge: Transforming the face of computer science through public high school education. *Learning, Media and Technology, 38*(2), 161–181. https://doi.org/10.1080/17439884.2013.756514

Sáez-López, J.-M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two-year case study using "Scratch" in five schools. *Computers & Education, 97*, 129–141. https://doi.org/10.1016/j.compedu.2016.03.003

Sherwood, H., Yan, W., Liu, R., Martin, W., Adair, A., Fancsali, C., Rivera-Cash, E., Pierce, M., & Israel, M. (2021). Diverse Approaches to School-wide Computational Thinking Integration at the Elementary Grades: A Cross-case Analysis. *SIGCSE - Proceedings of ACM Technical Symposium on Computer Science Education* (pp. 253–259). https://doi.org/10.1145/3408877.3432379

Shute, V. J., Sun, Ch., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review, 22*, 142–158. https://doi.org/10.1016/j.edurev.2017.09.003

Snodgrass, M. R., Israel, M., & Reese, G. C. (2016). Instructional supports for students with disabilities in K-5 computing: Findings from a cross-case analysis. *Computers & Education, 100*, 1–17. https://doi.org/10.1016/j.compedu.2016.04.011

Sun, L., Hu, L., Yang, W., Zhou, D., & Wang, X. (2021). STEM learning attitude predicts computational thinking skills among primary school students. *Journal of Computer Assisted Learning, 37*(2), 346–358. https://doi.org/10.1111/jcal.12493

Sysło, M. M. (2014). The First 25 Years of Computers in Education in Poland: 1965 – 1990. In A. Tatnall & B. Davey (Eds.), *Reflections on the History of Computers in Education: Early Use of Computers and Teaching about Computing in Schools* (pp. 266-290). Springer. https://doi.org/10.1007/978-3-642-55119-2_18

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers & Education, 148*. https://doi.org/10.1016/j.compedu.2019.103798

Taslibeyaz, E., Kursun, E., & Karaman, S. (2020). How to Develop Computational Thinking: A Systematic Review of Empirical Studies. *Informatics in Education, 19*(4), 701–719. https://doi.org/10.15388/INFEDU.2020.30

Taylor, R. D., Oberle, E., Durlak, J. A., & Weissberg, R. P. (2017). Promoting Positive Youth Development Through School-Based Social and Emotional Learning Interventions: A Meta-Analysis of Follow-Up Effects. *Child Development, 88*(4), 1156–1171. https://doi.org/10.1111/cdev.12864

Tikva, C., & Tambouris, E. (2021). Mapping computational thinking through programming in K-12 education: A conceptual model based on a systematic literature Review. *Computers & Education, 162*. https://doi.org/10.1016/j.compedu.2020.104083

Tsai, M.-J., Liang, J.-C., & Hsu, C.-Y. (2020). The Computational Thinking Scale for Computer Literacy Education. *Journal of Educational Computing Research, 59*(4), 579–602. https://doi.org/10.1177/0735633120972356

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. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (pp. 605–611). Association for Computing Machinery. https://doi.org/10.1145/3328778.3366809

Vuorikari, R., Punie, Y., Carretero Gomez S., & Van den Brande, G. (2016). *DigComp 2.0: The digital competence framework for citizens. Update phase 1: the conceptual reference model*. Publications Office of the European Union. https://doi.org/10.2791/11517

Waite, J., Curzon, P., Marsh, W., & Sentance, S. (2020). Difficulties with design: The challenges of teaching design in K-5 programming. *Computers & Education, 150*.

https://doi.org/10.1016/j.compedu.2020.103838

Webb, M., Davis, N., Bell, T., Katz, Y., Reynolds, N., Chambers, D. P., & Sysło, M. M. (2017). Computer science in K-12 school curricula of the 21st century: Why, what and when? *Education and Information Technologies, 22*(2), 445–468. https://doi.org/10.1007/s10639-016-9493-x

Wei, X., Lin, L., Meng, N., Tan, W., Kong, S.-C., & Kinshuk. (2021). The effectiveness of partial pair programming on elementary school students' Computational Thinking skills and self-efficacy. *Computers & Education, 160*. https://doi.org/10.1016/j.compedu.2020.104023

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., et al. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology, 25*(1), 127–147. https://doi.org/10.1007/s10956-015-9581-5

Weintrop, D., & Wilensky, U. (2015). Using Commutative Assessments to Compare Conceptual Understanding in Blocks-based and Text-based Programs. In *Proceedings of the eleventh annual International Conference on International Computing Education Research (ICER '15)* (pp. 101–110). Association for Computing Machinery. https://doi.org/10.1145/2787622.2787721

Wiliam, D. (2011). What is assessment for learning? *Studies in Educational Evaluation, 37*(1), 3-14. https://doi.org/10.1016/j.stueduc.2011.03.001

Wing, J. (2017). Computational Thinking's Influence on Research and Education for All. *Italian Journal of Educational Technology, 25*(2), 7–14. https://doi.org/10.17471/2499-4324/922

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society, 366*(1881), 3717–3725. https://doi.org/10.1098/rsta.2008.0118

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

Witherspoon, E. B., Higashi, R. M., Schunn, C. D., Baehr, E. C., & Shoop, R. (2017). Developing Computational Thinking through a Virtual Robotics Programming Curriculum. *ACM Transactions on Computing Education, 18*(1), 1-20. https://doi.org/10.1145/3104982

Wu, S.-Y., & Su, Y.-S. (2021). Visual Programming Environments and Computational Thinking Performance of Fifth- and Sixth-Grade Students. *Journal of Educational Computing Research, 59*(6), 1075–1092. https://doi.org/10.1177/0735633120988807

Yadav, A., Good, J., Voogt, J., & Fisser, P. (2017). Computational Thinking as an Emerging Competence Domain. In M. Mulder (Ed.), *Competence-based Vocational and Professional Education, Technical and Vocational Education and Training: Issues, Concerns and Prospects, 23*, 1051–1067. https://doi.org/10.1007/978-3-319-41713-4_49

Yin, R. K. (2014). *Case study research - Design and methods (5th edition)*. SAGE Publications.

Yin, R. K. (2003). Designing case studies. In L. Maruster & M.J. Gijsenberg (eds.) *Qualitative Research Methods* (pp. 359–386). SAGE Publications.

Zhang, L., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. *Computers & Education, 141*. https://doi.org/10.1016/j.compedu.2019.103607

Zhang, L., Nouri, J., & Rolandsson, L. (2020). Progression Of Computational Thinking Skills In Swedish Compulsory Schools With Block-based Programming. In *Proceedings of the Twenty-Second Australasian Computing Education Conference (ACE'20)* (pp. 66–75). Association for Computing Machinery. https://doi.org/10.1145/3373165.3373173

Terminology list

Abstraction – The process of filtering out or ignoring the characteristics of patterns that we do not need for finding solutions so we can concentrate on those that we do.

Algorithm & Algorithmic Thinking – Algorithms are precise step-by-step plans or procedures to meet an end goal or to solve a problem; algorithmic thinking is the skill involved in developing an algorithm (Grover & Pea, 2018).

Algorithm design – Creating an ordered series of instructions for solving similar problems or for performing a task.

Automation – Having computers or machines do repetitive tasks.

Boolean logic – A way of evaluating the truth value of an expression, where the elements are either true or false, and expressions are made up of the basic operators: AND, OR, NOT.

Coding – The implementation of solutions in a particular computer programming language.

Computational Thinking (CT) (several definitions are presented):

- The thought process that involves solving problems and designing model systems by utilising Computer Science core concepts (Wing, 2008).
- The thought processes involved in formulating a problem and expressing its solution(s) in such a way that a computer-human or machine can effectively carry it out (Grover & Pea, 2018).
- The thought processes involved in formulating problems so that their solutions can be represented as computational steps and algorithms (Aho, 2012).
- Computational Thinking describes the thought processes entailed in formulating a problem so as to admit a computational solution involving abstraction, algorithmic thinking, automation, decomposition, debugging and generalisation (2016 EU Computational Thinking study, Bocconi et al., 2016 p. 18).
- Recognising aspects of computation in the world that surrounds us and applying tools and techniques from Computer Science to understand, reason and solve problems in relation to both natural and artificial systems and processes (Webb et al., 2017).

Computer Science (CS) – The scientific discipline covering principles such as algorithms, data structures, programming, systems architecture, design, and problem-solving.

Computing – The domain incorporating Information Technology, Computer Science and digital literacy. Computing is also the name of a specific school subject in the UK-England curriculum. Note that in the context of the EC DEAP 2021-2027, Computer Science, Informatics, and computing are used as synonyms.

Conditional logic – Finding the associated pattern between different events (Grover & Pea, 2013).

Condition – In computing, this is a statement that is either true or false. A computation depends on whether a condition equates to true or false.

Conditionals – Making decisions based on conditions (Ching et al., 2018).

Data – A sequence of one or more symbols given meaning by specific acts of interpretation. Data can be analysed or used in an effort to gain knowledge or make decisions.

Data analysis – Storing, retrieving and updating values. Making sense of data by finding patterns or developing insights (Basu et al., 2017).

Data collection – A systematic process of gathering observations or measurements.

Data representation – Depicting and organising data in appropriate graphs, charts, words, or images.

Data structure – A particular way of organising data in a computer so that it can be used effectively.

Debugging & error detection – Finding your own mistakes and fixing them (Atmatzidou & Demetriadis, 2016).

Decomposition – Breaking down data, processes or problems into smaller, manageable parts. "Decomposition is used when a problem is too big or complex to solve at once, and when we know how to solve the subproblems effectively" (Rijke et al., 2018).

Digital competence – A set of skills, knowledge and attitudes that enable the confident, creative and critical use of information technologies and systems.

Digital content – Any type of content that exists in the form of digital data that are encoded in a machine-readable format, and can be created, viewed, distributed, modified and stored using computers and digital technologies.

Digital environment – A context or "place" that is enabled by technology and digital devices, often transmitted over the Internet, or other digital means, e.g., mobile phone network. Records and evidence of an individual's interaction with a digital environment constitute their digital footprint.

Digital literacy – The general ability to use computers. This is written in lower case to emphasise that it is a set of skills rather than a subject in its own right (The Royal Society, 2012).

Digital technology – Any product that can be used to create, view, distribute, modify, store, retrieve, transmit and receive information electronically in a digital form, for example, computers and mobile devices, digital television, robots.

Digital tools – Tools used for a given purpose or for carrying out a particular function of information processing, communication, content creation, safety or problem solving.

Efficiency & performance – Analysing the degree to which the solution meets requirements, often in terms of the times and resources employed in order to achieve better results (Grover & Pea, 2018).

Evaluation – A process that allows us to make sure our solution does the job it has been designed to do and to think about how it could be improved.

Event – One thing causing another thing to happen (Ching et al., 2018).

Function – A section of a program which performs a specific task that can be called up by another part of the program with the purpose of returning one single value.

Generalisation – A way of solving problems by drawing on previous solutions, thus building on prior experience; involves identifying patterns, similarities and connections, and exploiting these.

Informatics – The entire set of scientific concepts that make information technology possible (Europe Informatics, 2013).

Information Technology (IT) – The use of computers in industry, commerce, the arts and elsewhere, including use of software, aspects of systems architecture, human factors, project management, etc. This sense is narrower than its use in industry, which it generally encompasses Computer Science as well (The Royal Society, 2012).

Logical reasoning – A form of thinking in which premises and relations between premises are used in a rigorous manner to infer conclusions that are entailed (or implied) by the premises and the relations.
Logic & Logical thinking – analysing situations to decide on or reach a conclusion about a situation (Grover & Pea, 2018).

Loops – Running the same sequence multiple times.

Modelling – Developing a model to imitate real-world processes. In computing, modelling is used to examine large amounts of data to help with scientific or engineering projects. Solve a problem at hand through the model architecture or develop a new system.

Operators – Supporting mathematical and logical expressions (Ching et al., 2018).

Parallelisation – Simultaneous processing of smaller tasks from a larger task to reach a common goal more efficiently.

Pattern Generalisation – Creating models, rules, principles, or theories of observed patterns to test predicted outcomes.

Pattern Recognition – Identifying and observing patterns, trends, and regularities.
Procedure – A section of a program that performs a specific task.

Program – Sequences of instructions for a computer.

Programming – A process involving analysis and understanding of problems; identifying and evaluating possible solutions; generating algorithms; implementing solutions in the code of a particular programming language; testing and debugging, in order to formulate solutions into executable computer programs (Webb et al., 2017).

Problem solving – An individual's capacity to engage in cognitive processing to understand and resolve problem situations where a method of solution is not immediately obvious. It includes the willingness to engage with such situations in order to achieve one's potential as a constructive and reflective citizen (OECD, 2014).
Sequence – Creating a series of individual steps or instructions that can be executed by the computer (Ching et al., 2018).

Simulation – The process of modelling, performed on a computer, which is designed to predict the behaviour of a real-world or physical system.

Statement – The smallest element of a programming language which expresses an action to be carried out.

Transformation – Conversion of a collection of information (Wing, 2006).

Unplugged activities – Work performed without the use of computer technology.

Variable – A memory location within a computer program where values are stored.

Visualisation – The representation of an object, situation, or dataset in graphic form, e.g., chart, graph, etc.

List of abbreviations

CS	Computer Science
CT	Computational Thinking
CTEPI	Computational Thinking Educational Policy Initiatives
EU	European Union
GSCE	General Certificate of Secondary Education (school qualification in England, Wales & Northern Ireland)
ICT	Information and Communication Technologies
ISCED 1	International Standard Classification of Education – Level 1 (primary education)
ISCED 2	International Standard Classification of Education – Level 2 (lower-secondary education)
ISCED 3	International Standard Classification of Education – Level 3 (upper-secondary education)
ITE	Initial Teacher Education (pre-service training of aspiring teachers)
iVET	Initial Vocational Education & Training (commenced in lower secondary school)
JRC	Joint Research Centre
MCS	Multiple-Case Study
NCC	National Computing Centre (UK)
OECD	Organisation for Economic Co-operation and Development
SEN	Special Education Needs
UNESCO	United Nations Educational, Scientific and Cultural Organisation
VET	Vocational Education & Training

Country codes

EU	European Union
AT	Austria
BE	Belgium
BE fr	Belgium - French Community
BE nl	Belgium - Flemish Community
CH	Switzerland
CY	Cyprus
CZ	Czech Republic
DK	Denmark
EL	Greece
ES	Spain
FI	Finland
FR	France
HR	Croatia
HU	Hungary
IE	Ireland
IL	Israel
IT	Italy
LT	Lithuania
LU	Luxembourg
MT	Malta
NO	Norway
PL	Poland
PT	Portugal
RO	Romania
RS	Serbia
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
UK-ENG	England

List of figures

List of boxes

List of tables

Annex 1. Ministries of Education and other educational institutions contributed to the survey

Country	Organisation
Austria	Federal Ministry of Education, Science and Research
Belgium	Flemish Ministry of Education
	Service général de l'inspection
Croatia	Ministry of Science and Education
Cyprus	Ministry of Education, Culture, Sport and Youth
Czech Republic	Czech National Agency for International Education and Research (Dům zahraniční spolupráce)
Denmark	The Danish Agency for Education and Quality (Styrelsen for Undervisning og Kvalitet)
Finland	The Finnish National Agency for Education
France	French Ministry of Education, Youth and Sports
Greece	Ministry of Education
Hungary	Educational Authority / Oktatási Hivatal
Ireland	National Council for Curriculum and Assessment (NCCA)[97]
Italy	Indire
Lithuania	National Agency for Education
Luxembourg	IFEN (Institut de l'éducation nationale)
Malta	Ministry for Education Malta
Poland	Ministry of Education and Science
Portugal	General Directorate for Education
Romania	National Centre for Policies and Assessments in Education, Ministry of Education
Slovakia	Comenius University
Slovenia	Ministry of Education, Science and Sport
Spain	INTEF - Ministry of Education and VET

Country	Organisation
Georgia	Ministry of Education and Science of Georgia
Israel	Ministry of Education
Norway	Norwegian Directorate for Education and Training
Russia	The Russian Academy of Natural History
Serbia	Tempus Foundation
Singapore	National Institute of Education
Switzerland	Swiss Conference of Cantonal Ministers of Education EDK

97. The National Council for Curriculum and Assessment (NCCA) is a statutory body of the Department of Education

Annex 2. Participants to the online consultation events

Topic 1: CT understanding

Name	Surname	Country	Affiliation
Augusto	**Chioccariello**	IT	CNR-ITD
Anusca	**Ferrari**	BE	DG EAC
Laura	**Freina**	IT	CNR-ITD
Nikoleta	**Giannoutsou**	ES	JRC
Eglė	**Jasutė**	LT	Vilnius University
Ken	**Kahn**	UK	University of Oxford
Noa	**Ragonis**	IL	Beit Berl Colege
Sue	**Sentance**	UK	Raspeberry Pie
Matti	**Tedre**	FI	University of Eastern Finland
Aman	**Yadav**	USA	Michigan State University

Topic 2: CT assessment

Name	Surname	Country	Affiliation
Romina	**Cachia**	ES	JRC
Valentina	**Dagienė**	LT	Vilnius University
Birgit	**Eickelmann**	DE	University of Paderborn, Germany
Shuchi	**Grover**	UK	Stanford university
Chee-Kit	**Looi**	SG	National Institute of Education, Singapore
Chiara	**Malagoli**	IT	CNR-ITD
Vaida	**Masiulionytė-Dagienė**	LT	Vilnius University
Jesus	**Moreno León**	ES	Universidad Rey Juan Carlos
Alexander	**Repenning**	CH	University of Applied Sciences and Arts Northwestern Switzerland (PH FHNW)

Topic 3: Quality computing education

Name	Surname	Country	Affiliation
Stefania	**Bocconi**	IT	ITD
Ania	**Bourgeois**	BE	EACEA
Katja	**Engelhardt**	BE	EUN
Milena	**Horvath**	BE	EUN
Vania	**Nieto**	BE	Microsoft
Veronica	**Mobilio**	BE	DG EAC
Tarek	**Mostafa**	FR	OECD, Directorate for Education and Skills
Enrico	**Nardelli**	IT	University of Rome "Tor Vergata"
Pat	**Yongpradit**	USA	Chief Academic Officer at Code.org

Topic 4: CT in iVET

Name	Surname	Country	Affiliation
Jeffrey	**Earp**	AUS / IT	CNR-ITD
Andreia	**Inamorato**	ES	JRC
Panagiotis	**Kampylis**	EL	CNR-ITD
Daniel	**Krupka**	DE	German Informatics Society
Paulo	**Moekotte**	NL	Senior beleidsadviseur Onderwijs en Kwaliteitszorg
Alexander	**Romiszowski**	USA	Middlesex Polytechnic in the UK and Syracuse University in the USA
Christian	**Schrack**	AT	Federal Ministry of Education
Gabrielė	**Stupurienė**	LT	Vilnius University
Stefano	**Tirati**	IT	EfVET Network
Georgi	**Dimitrov**	BE	DG EAC
Ioannis	**Maghiros**	ES	JRC
Yves	**Punie**	ES	JRC

Topic 1: Understanding of CT and related concepts

Name	Surname	Country	Affiliation
Eleni	**Berki**	FI	University of Jyväskylä
Filomena	**Faiella**	IT	University of Salerno
Gerald	**Futschek**	AT	Vienna University of Technology
Mikko-Jussi	**Laakso**	FI	Centre for Learning Analytics, University of Turku
Simone	**Opel**	DE	Fernuniversität in Hagen, Fak. f. Mathematik Informatik / GI e.V.
Tauno	**Palts**	EE	University of Tartu
Don	**Passey**	UK	Lancaster University
Noa	**Ragonis**	IL	Beit Berl College
Maciej	**Sysło**	PL	Torun University
Valentina	**Dagienė**	LT	Vilnius University
Gabrielė	**Stupurienė**	LT	Vilnius University
Vaida	**Masiulionytė-Dagienė**	LT	Vilnius University
Nikoleta	**Giannoutsou**	EU	CNR-ITD

Topic 2: Major trends in CT integration within compulsory education

Name	Surname	Country	Affiliation
Erik	**Barendsen**	NL	Radboud University & Open University
Olga	**Davydovskaia**	BE	Education and Culture Executive Agency, Eurydice Unit
Natasa	**Grgurina**	NL	Stichting leerplanontwikkeling
Yasemin	**Gulbahar**	TR	Ankara University
Juraj	**Hromkovic**	CH	ETH Zurich
Dennis	**Komm**	CH	PH Graubünden
Amelie	**Labusch**	DE	Paderborn University
Zsuzsa	**Pluhár**	HU	Eötvös Loránd University
Pohl	**Wolfgang**	DE	Bundesweite Informatikwettbewerbe
Katja	**Engelhardt**	BE	EUN
Milena	**Horvath**	BE	EUN
Eglė	**Jasutė**	LT	Vilnius University
Andreia	**Inamorato**	EU	JRC

Topic 3: Approaches to CT teaching, learning and assessment

Name	Surname	Country	Affiliation
Jüliyet	**Bahar**	SE	Vendelsömalmsskolan
Vaino	**Brazdeikis**	LT	National Agency of Education
Katarina	**Grgec**	HR	Ministry of science and education
Deirdre	**Hodson**	EU	DG EAC
Joonas	**Meri**	FI	Armfeltin koulu (lower secondary school)
Alexander	**Repenning**	CH	University of Applied Sciences and Arts Northwestern Switzerland
Michal	**Rybár**	SK	Ministry of Education, Slovakia
Morten	**Søby**	NO	Norwegian Directorate for Education and Training
Małgorzata	**Szybalska**	PL	Ministry of Education and Science
Daniel	**Toms**	UK	Harrogate Grammar School
Augusto	**Chioccariello**	IT	CNR-ITD
Jeffrey	**Earp**	AUS / IT	CNR-ITD
Chiara	**Malagoli**	IT	CNR-ITD
Romina	**Cachia**	EU	JRC

Topic 4: Implications for policy and practice

Name	Surname	Country	Affiliation
Ania	**Bourgeois**	BE	EACEA
Patrick	**Camilleri**	MT	University of Malta
Jan	**De Craemer**	BE	Flemish Ministry of Education and Training
José Luis	**Fernández**	ES	Ministry of Education and Vocational Training
Vibeke	**Guttormsgaard**	NO	Norwegian Directorate for Education and Training
Daniel	**Krupka**	DE	German Informatics Society
Coromoto	**León**	ES	Universidad de La Laguna
Enrico	**Nardelli**	IT	Informatics Europe
Annika	**Östergren Pofantis**	BE	European Commission - EU Code Week
Patricia	**Wastiau**	BE	EUN
Stefania	**Bocconi**	IT	CNR-ITD
Panagiotis	**Kampylis**	EL	CNR-ITD
Anusca	**Ferrari**	EU	DG EAC
Yves	**Punie**	EU	European Commission - DG JRC

Annex 3. Contributors of the oral and written interviews[98]

Multiple Case Study 1

Name	Surname	Organisation	Country	Role
Ivan	**Kalaš**	Comenius university	Slovakia	Expert for MCS1
Vaino	**Brazdeikis**	National Agency for Education	Lithuania	Policy maker
Aurelijus	**Liaudanskas**	Klaipėdos Gedminų progimnazija	Lithuania	School leader
Indra	**Sudeikienė**	Klaipėdos Gedminų progimnazija	Lithuania	Teacher
7 students		Klaipėdos Gedminų progimnazija	Lithuania	Grade 4 students
Vibeke	**Guttormsgaard**	Directorate of Education and training	Norway	Expert
Ingvild	**Vikingsen Skogestad**	Knappskog school	Norway	School leader
Øyvind	**Rise**	Knappskog school	Norway	Teacher
6 students		Knappskog school	Norway	Grade 4 students
Michal	**Rybár**	Ministry of Education, Science, Research and Sport of the Slovak Republic	Slovakia	Policy maker
Zlatica	**Lišková**	The Ludovit Stur Primary School	Slovakia	School leader
Jana	**Nemčovičová**	The Ludovit Stur Primary School	Slovakia	Teacher
6 students		The Ludovit Stur Primary School	Slovakia	Grade 5 students

Multiple Case Study 2

Name	Surname	Affiliation	Country	Role
Peter	**Hubwieser**	Technical University of Munich	Germany	Expert for MSC2
Katarina	**Grgec**	Ministry of Science and Education	Croatia	Head of sector
Lidija	**Kralj**	European Schoolnet	Croatia	Education analyst & advisor
Dražen	**Mlakar**	Primary School Popovača	Croatia	School head
Darko	**Rakić**	Primary School Popovača	Croatia	Teacher
5 students		Primary School Popovača	Croatia	Grade 8 students
Małgorzata	**Szybalska**	Ministry of Education and Science	Poland	General counsel
Maciej M.	**Sysło**	Warsaw School of Computer Science	Poland	Professor
Dariusz	**Andrzejewski**	Samorządowa Szkoła Podstawowa nr 6	Poland	School head
Dorota	**Czech-Czerniak**	Samorządowa Szkoła Podstawowa nr 6	Poland	Teacher
6 students		Samorządowa Szkoła Podstawowa nr 6	Poland	Grade 8 students
Simon	**Peyton Jones**	Microsoft Research	UK-England	Principal researcher
Neil	**Renton**	Harrogate Grammar School	UK-England	School head
Daniel	**Toms**	Harrogate Grammar School	UK-England	Teacher
2 students		Harrogate Grammar School	UK-England	Grade 8 students

98. The following native speakers provided support and translation services for the interviews with teachers and students: Lidija Kralj (Croatian), Juri Valtanen and Eleni Berki (Finnish), Patricia Wastiau (French), Eglė Jasutė, Gabrielė Stupurienė and Vaida Masiulionytė-Dagienė (Lithuanian), Ingvild Vikingsen Skogestad (Norwegian), Ola Miklasińska (Polish), Ivan Kalaš (Slovakian), and Helena Isaksson Persson (Swedish).

Multiple Case Study 3

Name	Surname	Organisation	Country	Role
Celia	**Hoyles**	University College London	United Kingdom	Expert for MCS3
Françoise	**Tort**	ENS Paris Saclay	France	Expert
Jean	**Bataille**	Collège Vauban de BELFORT	France	School head
Philippe	**Tissot**	Collège Vauban de BELFORT	France	Maths teacher
David	**Balaud**	Collège Vauban de BELFORT	France	Tech teacher
5 students		Collège Vauban de BELFORT	France	Students
Isabelle	**Salome**	College Alberto Giacometti Montigny Le Bretonneux	France	School head
Martine	**Roux**	College Alberto Giacometti Montigny Le Bretonneux	France	Maths teacher
5 students		College Alberto Giacometti Montigny Le Bretonneux	France	Students
Matti	**Ranta**	Finnish National Agency for Education (OPH)	Finland	Policy maker
Matti	**Laakso**	Turku university	Finland	Expert
Joonas	**Mori**	Armfelt School	Finland	Maths teacher
Jaana	**Talvitie**	Armfelt School	Finland	School head
Jouko	**Järvenpää**	Armfelt School	Finland	Maths/Crafts teacher
5 students		Armfelt School	Finland	Grade 9 students
Christian	**Magnusson**	Swedish Ministry of Education	Sweden	Policy maker
Mats	**Hansson**	Skolverket	Sweden	Policy maker
Johan	**Falk**	Skolverket	Sweden	Expert
Ulrike	**Petterson**	Vendelsomalmsskolan	Sweden	School head
Juliyet	**Bahar**	Vendelsomalmsskolan	Sweden	Maths teacher
Susanna	**Albinsson**	Vendelsomalmsskolan	Sweden	Tech teacher
5 students		Vendelsomalmsskolan	Sweden	Grade 9 students

Experts contributed to the written interviews for the integration of CT in iVET within compulsory education

Name	Surname	Organisation	Country	Role
Daniel	**Krupka**	German Informatics Society	Germany	Managing director
Simon	**Opel**	German Informatics Society	Germany	Spokesperson of expert group on vocational training
Paulo	**Moekotte**	ROC van Twente	The Netherlands	Senior education Policy advisor
João	**Piedade**	Institute of Education of University of Lisbon	Portugal	Assistant professor

Annex 4. Multiple case studies

Key findings from Multiple Case 1: Computational Thinking as *cross-curricular theme* in primary education

This Multiple-Case Study (MCS) looks at how Computational Thinking (CT) is implemented in primary education (ISCED 1) in Lithuania, Slovakia, and Norway. Lithuania and Slovakia have a long-standing tradition in Computer Science education, which is not the case in Norway. However, Lithuania and Norway have started introducing Computational Thinking as an integrated part of primary schooling, while Slovakia has implemented Computational Thinking in a separate Informatics subject in grade 3. Therefore, contrasting the cases of Lithuania and Norway with the case of Slovakia can yield interesting insights on the matter of CT integrated as a cross-curriculum theme versus adoption of CT in a separate subject in primary education.

Computational Thinking conceptualisation and understanding – MSC1

Lithuania explicitly declares introduction of Computational Thinking in the general primary curriculum[99] and expresses understanding of the concept by describing students' abilities to work with data and models, to solve problems and automate solutions. Lithuania's primary school Informatics curriculum describes Computational Thinking as the ability to identify, formulate and solve problems (tasks), logically organize and analyse data, apply schemes and models, develop the capability to resolve problems algorithmically and logically, and automate the solution using digital technologies[100].

In 2020 Norway prepared a new curriculum concerning all subjects. Its overarching objective is to enable children and young people to meet and find solutions to the challenges of today and tomorrow. In Norway, Computational Thinking is understood as algorithmic thinking and is part of Programming, but in primary education programming is also integrated within four other subjects: Mathematics, Science, Art & Crafts, and Music.

In the Slovakian view, Informatics and Computational Thinking mean understanding the basic concepts behind computers and digital technologies; basically, the definition can be expressed as the thinking of a computer or a digital device and is deeply connected with programming. Teaching programming is spreading and improving rapidly as a primary school topic in Slovakia. Its introduction as an integral part of the Informatics subject has been carefully designed and piloted in schools. This initiative addresses all primary teachers and their students (every learner in the class). In general, primary school teachers use expressions like algorithm, programming, computational thinking, sequence of steps, robotics, repetition/loop, procedure, logical thinking, mathematical thinking, and reasoning.

Implementation of Computational Thinking at primary school level – MSC1

In Lithuania, informatics is taught as a part of other subjects in primary education. It was general agreed to introduce it gradually, starting from innovative schools and teachers. The country's Ministry of Education, Science and Sport approved Guidelines for Updating the General Curriculum Framework in 2019, and 100 primary schools have been officially selected to implement the new Informatics curriculum since then. However, in the General Teaching Plan for 2021 and 2022, all primary schools are encouraged to implement Informatics in their classes (pp. 29, clause 82.6.2).[101] Lithuanian educational institutions are currently organising consultation and training programmes for primary school teachers to support the introduction of this innovation. While this new primary education Informatics curriculum covers most of the major Computational Thinking components, it is up to individual schools (and their teachers) to decide how and in which subjects to

99. https://www.e-tar.lt/portal/lt/legalAct/e3e9269009e511ea9d279ea27696ab7b (clauses 49-50)
100. https://www.emokykla.lt/bendrasis/bendrosios-programos/bendrojo-ugdymo-programu-projektai-2021-11-03
101. https://www.nsa.smm.lt/2021/06/28/2021-2022-ir-2022-2023-mokslo-metu-pradinio-pagrindinio-ir-vidurinio-ugdymo-programu-bendrieji-ugdymo-planai

integrate the Informatics components they choose to address.

The revised Norwegian national curriculum came into force in 2020. The country's Ministry of Education published a digitalization strategy for primary, secondary, and vocational education for 2017-2021, stating that Computational Thinking and programming should be integrated into the curriculum. The new curriculum foresees the integration of Computational Thinking and programming in Maths, Science, Arts & Crafts, and Music, both at primary and lower secondary levels. The curriculum emphasizes the development of algorithmic thinking skills by addressing the steps involved in solving problems through programming.

Slovakia's comprehensive reform of its education system in 2008 brought significant changes to the content of all subjects in primary and secondary schools. One of the main changes was the introduction of Informatics as a compulsory separate subject in Grades 3-4[102]. Students are introduced to basic computational concepts starting from simple computer instructions and then arranging these instructions into procedures and loops.

In all three countries, Computational Thinking is being implemented gradually in primary education, mostly hinging on teaching programming and introducing basic computational concepts.

Pedagogical approaches, methods, and tools – MSC1

In Lithuania, Teachers choose the methods according to their experience and intuition, and to the classroom milieu at the time. Lithuanian students use various tools: Xlogo, Imagine Logo, Scratch, Ville/Eduten (a virtual learning environment developed by Turku University, Finland[103]), CodeMonkey, Bebras tasks, Bebras task cards (https://bebras.lt), Logo, Blue Bot, EMA (national digital tool), StoryJumper. An interviewed teacher summarized that Bebras tasks, ViLLE, XLogo, and Scratch are the most commonly used of these.

In Norway, teacher tutorials are provided to help implement schools' coding plans. Most teachers adopt learning by doing, learning from mistakes, and share their experiences with other teachers. Class-to-class progression in coding plans is well established and flexibly applied. Students' peer-to-peer collaboration and support are encouraged.

In Slovakia, an essential part of the Informatics curriculum is dedicated to age-appropriate programming for each grade. Tools such as a specially designed educational program called Emil, as are Pro-Bot and Blue-Bot robots. In primary years, programming is connected to other subjects in the program. For introducing Computational Thinking, Bebras Challenge[104] (known in Slovak iBobor[105]) is used. Up to 30% of all schools run iBobor activities.

In Slovakia there is support both for integrating CT skills development as a cross-curricula theme and in a separate CS subject in primary education.

All three countries use a variety of educational tools for developing skills in Computational Thinking/Informatics/programming. Some are from abroad, while others have been specially designed and promoted by national institutions. Learning by doing is the most commonly adopted methodology.

CT assessment – MSC1

CT assessment is based on teacher observation/monitoring. In Slovakia this is performed generally, while in Norway is it undertaken especially in group discussion and project work; in Lithuania it is carried out while learners are working on problems, exercises or quizzes, using different digital tools. Learning by mistake-making is emphasised in Norway.

102. https://eacea.ec.europa.eu/national-policies/eurydice/content/teaching-and-learning-single-structure-education-30_en
103. Ville/Eduten: https://www.oppimisanalytiikka.fi/ville/
104. International Bebras Challenge on Informatics and Computational Thinking: https://www.bebras.org
105. Bebras in Slovakia: https://ibobor.sk

In Lithuania, indications on pupil assessment and achievement are set out in the General Curriculum for Primary Education and in the General Plan for the Primary Education Curriculum for the relevant school year[106]. Learners' achievements in primary school are recorded as short commentaries and descriptions. Grades/marks (scores) or signs and symbols are not used. Assessment in Informatics is mainly implemented through projects and formative assessments. Teachers also set quizzes, exercises, and surveys. Each school stipulates the methods and procedures for assessing students' learning achievements and progress.

In Slovakia, assessment involves marking (on a scale from 1 to 5) performed through monitoring of learner performance and preparedness for lessons. In Informatics pupils mainly work in teams, discussing what they think, how they think, and how they come up with solutions. Teachers' assessment is done through monitoring learners and also through reading and commenting on their solutions.

In Lithuania and Norway, assessment at the primary stage does not involve the awarding of marks, while in Slovakia a marking scale from 1 to 5 is used. However, interviewees from all these countries emphasised that the monitoring of student learning during classes is a very important method.

Gender and equity issues – MSC1

Lithuania has no specific strategy to ensure gender balance and equity at primary school level. In Norway gender balance was one of the reasons for making Computational Thinking and programming compulsory in primary schools. No issue is perceived in Slovakian for the same reason.

Teacher recruitment and professional development – MCS1

In general, in-service teacher training activities are carried out by universities, municipalities, agencies, centres and schools. First and foremost, it is dedicated to encouraging and motivating teachers to teach Informatics and developing learners' Computational Thinking skills. Improving teachers' digital competences also remains an important issue. In accordance with teachers' needs, institutions hold training seminars, invite lecturers or, at teachers' request, provide professional development courses in Informatics education. Interviewees stressed that even teachers with a computing background are not sufficiently prepared to teach Informatics or programming to young children.

Norway has not recruited new teachers because primary school teachers teach almost all subjects. A five-module Competence Package for primary school has been developed to give teachers an understanding of what programming is and how they can work with programming in their subjects, based on the new curriculum.

Interviewees agree that primary school teachers should be constantly updated because almost all material quickly becomes outdated in the Informatics field.

Lessons learnt – MCS1

Main drivers

Ministries of Education, universities and other institutions participated actively in implementing the Informatics curriculum in primary education because they are interested in future generation digital competences and imparting a deeper understanding of technologies.

Slovakia and Lithuania approach Computational Thinking as a cross-curriculum theme in grades 1 and 2 (probably also in pre-school education) and implement it in a separate Informatics subject in grades 3 and 4. Teaching Computational Thinking as a topic at primary school is considered vital for students to understand the digital world around them and to be technological adaptive and creative when solving problems.

One of the main general conclusions from MCS1 is that Computational Thinking/Informatics/Programming is becoming an integral part of the primary school

106. https://www.nsa.smm.lt/2021/06/28/2021-2022-ir-2022-2023-mokslo-metu-pradinio-pagrindinio-ir-vidurinio-ugdymo-programu-bendrieji-ugdymo-planai

curriculum, whether as a cross-curriculum theme or included in a separate Informatics subject.

Main Challenges

The main challenge is the shortage of adequately trained teachers. Expert teachers are more likely to be able to motivate pupils for deeper learning of Informatics in the future. The emphasis is less on the Informatics curriculum itself but on how it is implemented and how it is understood. As a result, more focus is placed on the practical aspects and less on the basic principles and concepts, which however are very important both for teachers and learners alike.

Policy recommendations

- For implementing Computational Thinking in primary education, a mixed approach can be recommended: integration as a cross-curriculum theme in the early stages of primary school (grades 1 and 2) and incorporation in a separate Informatics subject from grades 3-4 (Lithuania and Slovakia).
- Integration of Computational Thinking can contribute to gender balance and to fostering students' creativity, with educational institutions paying more attention to training young people to be creators, not just consumers.
- The implementation of teaching Informatics at primary level requires collaborative work – it is essential to consult and re-negotiate with teachers, parents, and the community. If enthusiasm is wide-spread, a new challenge will be more readily accepted, and progress will be made.
- Include the basics of Informatics in the pre-service education of primary school teachers.
- Provide quality methodological support to in-service primary school teachers, including regular training, sharing of innovative pedagogical approaches, and proven time-thematic plans.

Key findings from Multiple Case 2: CT as *a separate subject* in lower secondary education

This MCS looks at how CT is integrated as part of a separate CS subject in Croatia (Kralj, 2016), Poland and the UK-England. This separate subject is called "Computing" in England, "Computer Science" in Poland and "Informatics" in Croatia. In recent years UK-England has been in the vanguard, being one of the first countries in Europe to mandate Computer Science as a foundational subject in primary and secondary schools as of 2014. Poland introduced their current CS curriculum in 2017[107] and Croatia in 2018.[108]

One hypothesis at the outset of MSC2 was that there might be differences in how CS curricula are implemented in Poland and Croatia, which both have a long-standing tradition in CS (Sysło, 2014), unlike England. In Croatia Informatics has been part of the curriculum since the 1990s[109]. UK-England's reputation in CS, however, started to decline in the late 1960s. The demise of the domestic computing industry corresponded with the declining quality of UK-England's computing education (Fowler & Vegas, 2021). The long-standing tradition in Croatia and Poland eased the transition to the new CS curriculum. In Croatia, CS was already a curriculum subject, so finding a place for it was obviated (this is a common challenge), and initial teacher training for CS was already in place. Both Poland and Croatia could rely on teachers who were already trained in CS. By contrast, in UK-England teachers had to be upskilled to teach the newly introduced Computing subject, constituting a huge challenge. Many ICT teachers in schools were language or maths teachers who were teaching students to use computers. So teacher training in the didactics of computing education was a challenge. In Poland and Croatia, introducing the current CS curricula has remained a challenge, since the current curricula CS has been comprehensively integrated at all education levels[110] and its focus has been adjusted. The new curriculum necessitated large scale training also for CS teachers. Finally, recruiting CS teachers remained – and

107. The current curriculum in Poland is in place since 2017 in primary schools (1st to 8th grade). With the school year 2019/2020 the reform also started in secondary schools and curricular changes are still work in the process of being implemented.
108. In Croatia, in January 2018, the preparation of teachers started and in September 2018 the new curriculum was implemented in grades 5-8 in primary school and grades 1-4 in secondary school (gymnasium). In September 2020, the new curriculum was implemented in grades 1-4 in primary school. Now all twelve years of general education in Croatia have a new Informatics curriculum.
109. Informatics was offered as an extra-curricular activity and a compulsory subject at upper secondary level in high schools and a majority of vocational programmes (and as an elective subject in grades 5-8 and extra-curricular activity in grades 1-4)
110. In Poland, the subject "Zajęcia komputerowe" (computer lessons) was previously compulsory in grades 1-3 and 4-6 and covered mainly ICT with some elements of informatics and robotics.

remains to be – a challenge also in these countries.

CT conceptualisation & understanding – MSC2

The definitions of CT in Poland and Croatia both place particular focus on problem solving. The curriculum in UK-England does not actually provide a definition of the term. The term CT is usually not the only term used in any of the countries. In Poland, for example, CT tends to be more used in policy documents or discussions with experts, while the term programming tends to be more commonly used by teachers and students. In UK-England, the term CT is more commonly used in primary schools, while the term programming is more widely used in secondary schools.

When it comes to defining the term CT, the Multiple-Case Study Expert expressed the position that much energy has been spent in the community on coming up with a commonly agreed definition – but without success. The expert considers as a more promising way forward to be discussing the integration of CT skills in CS curricula based on concrete teaching examples, a way to ensure shared understanding among stakeholders about the teaching and learning activities to be taught. Two of the Multiple-Case Study experts agree that while the term CT has the merit of attracting the interest of those outside the Computer Science community, it has two shortcomings: (i) it does not make clear enough that there is a lot of underlying proficient subject knowledge and a foundational subject discipline (e.g., understanding the concepts behind how complex things on the Internet work, how logical data structures function, the limitations of what computers can do); (ii) it does not fully recognise that CS should be considered as a foundational subject like Mathematics and languages. Several interviewees also emphasized that policy makers sometimes do not have a full and accurate understanding of what CS actually is, something which is necessary for defining the core components of a CS curriculum.

CT implementation at school level – MSC2

In all three countries, the CS subject covers both CS-related and digital competence/literacy related topics. For example, in the Croatian curriculum, activities related to CT skills take up 20-30% of teaching time in early grades, and up to 70% in upper secondary education. Similarly, in UK-England, the amount of time dedicated to CS concepts increases at higher grade levels. In UK-England, the Computing subject is compulsory for Key Stages 1 to 3 (until students are aged 14). As a matter of principle, the National Curriculum in England does not specify the number of hours to be taught per week. In Poland, one hour of CS as a separate subject is compulsory at all grades in K-12 education, while in Croatia, two hours are compulsory in grades 5 and 6. Generally, teachers still have some freedom to decide how much time they devote to which topic.

As regards the implementation of CS at school level, several "lessons learned" emerge. Poland and UK-England seem to fare well with curricula[111] that are rather open and not very prescriptive, aiming to grant teachers autonomy on how to implement the curricula. These curricula are accompanied by step-by-step guidelines[112], lesson plan examples and high-quality teaching resources for teachers in need of additional support. The CS curriculum in Croatia describes learning outcomes and levels of achievement in great details, which helps teachers to create content that is suitable for all students, including those with special needs and gifted students.

There seems to be a consensus that to teach CS concepts and related CT skills properly, these need to be taught as part of a separate subject, albeit not exclusively. However, there is also general agreement that skills acquired in CS education can foster learning in other subjects. Since one key rationale for teaching CS concepts and related CT skills is that they enable students to understand the digital world around them, key elements of the CS curriculum need to be compulsory for all.

111. Croatia: Kurikulum nastavnog predmeta Informatika za osnovne skole i gimnazije.pdf (gov.hr)
Poland: Podstawa programowa kształcenia ogólnego dla liceum, technikum i branżowej szkoły II stopnia – Ośrodek Rozwoju Edukacji (ore.edu.pl); England: National curriculum in England: computing programmes of study - GOV.UK (www.gov.uk)
112. Poland: Zestawy narzędzi edukacyjnych do wychowania przedszkolnego i kształcenia ogólnego - Zintegrowana Platforma Edukacyjna (zpe.gov.pl)
England: The essential toolkit for secondary computing teachers (teachcomputing.org)
Inspiration and support for teaching primary computing (teachcomputing.org)
Isaac Computer Science

The general trend seems to be to move towards making CS compulsory starting from early grades throughout the entire span of compulsory education. In several contexts, CS education is offered as a compulsory element in some grades and as an optional element in others.

Pedagogical approaches, methods, and tools – MSC2

With their often more open formulation and focus on fostering skills like problem-solving and logical thinking skills, CS curricula seem to lend themselves to pedagogical approaches that promote student autonomy. Examples of such approaches are personalised learning, project-based approaches and collaborative learning. The merits of the last of these were quite strongly emphasised not only by Case Study teachers but also by the students. Curricula in all three countries leave autonomy to teachers on how to implement concrete activities in their classrooms. These rather open curricula are complemented by step-by-step guidance, teaching resources, and examples for teachers who need more guidance.

Curricula in Poland and Croatia suggest teachers should start young students on visual languages and gradually progressi to text-based languages. The Polish curriculum does not prescribe a specific programming language but suggests a progression (e.g., from Code.org to Scratch, to Blockly, and then to Python and C++). Game-based approaches are favoured for younger students. The teachers in the Case Study schools in Poland and Croatia start their teaching by identifying their students' interests and needs. The curricula themselves do not prescribe which tools teachers are to use. Case Study teachers usually provide their students with choices between different tools.

What seems to be at the core of successful CS education is to enable students to work on real-life problems or create something on their own and let them find and correct errors on their own in the process. As a result, students get the chance to gain pride in their achievements, and teachers become guides-on-the-side lines, providing guidance and feedback. In conclusion, one could say that implementing the CS curricula in the way described above probably requires both teachers and their students to have courage and open-mindedness about the learning process and the results gained.

CT assessment – MSC2

Assessment in CS needs to be fair both to very able and less able students – it should capture excellence without discouraging average students. As part of active learning approaches described in the previous section, particular focus is devoted to formative assessment practices, which focus on providing feedback to students and helps them progress.

In Croatia, assessment in CS takes place via three assessment approaches: assessment for learning, assessment as learning, and assessment of what is learned. Guidelines[113] recommend focusing assessment on the following aspects: acquired knowledge, problem solving, digital contents, and collaboration. The "problem-solving" element includes assessing students on how they analyse and model problems, tackle steps, write algorithms, test the validity of algorithms, search and collect strategies, research, and construct a logic path. Autonomy in solving problems is a significant element.

In UK-England, as CS is part of GSCE exams, this creates some pressure on teachers to properly teach and assess CS. Due to COVID-19, assessments in the Computing subject conducted during the 2020/21 school year tested students' understanding of core concepts rather than requiring them to program full solutions to problems. This could be seen as more in line with general curriculum goals like fostering students' understanding of CS concepts, also since only 1.4% of all students actually opted for the GSCE exam in Computing in 2020[114]. Finally, according to the expert in UK-England, it is national high-stakes qualifications that

113. Smjernice za vrednovanje procesa učenja i ostvarenosti ishoda u osnovnoškolskome i srednjoškolskome odgoju i obrazovanje, Guidelines for assessment of learning process and learning outcomes achievement in primary and secondary education, Ministry of Science and Education, 2020
Pravilnik o načinima, postupcima i elementima vrednovanja učenika u osnovnoj i srednjoj školi, Policy on models, procedures and elements of student assessment in primary and secondary school (clean version), Ministry of Science and Education, 2019.
114. Computer Science in the Classroom Report – OKdo

determine the success of a CS curriculum. The expert would welcome attractive qualifications with a more applied focus in the UK.

Gender and equity issues – MSC2

Equity in CS education can refer to several dimensions such as gender balance, the mix of abled and less abled students or ethnic diversity. When looking at equity in CS education, one can consider several elements. The first is access to the CS subject. Where the subject is compulsory, equal access for all students is ensured. Only where CS is offered as an elective subject does the question of equal access potentially become an issue. In Croatia, CS is offered as a compulsory subject in grades 5 and 6. In grades 7 and 8, where the subject is offered as an elective, respectively 74% and 72.2% of students still opted to take CS in the 2020/21 school year.

The extent to which gender balance becomes an issue in such cases seems to differ between countries. In UK-England, out of the 78,459 students who opted to take Computer Science at GSCE level in 2020 only 18% were girls[115]. Interviewees of the Croatian Case Study claim that girls are well represented also at grades where CS is an elective subject. What all countries seem to have in common is the lack, at system level, of general data from all schools on the student population participating in CS as an elective (and with what degree of success). These data would be helpful for further investigating and addressing this issue.

The Case Studies seem to suggest that there might be another dividing line between generally more capable students and those with lower achievement levels. As the Croatian policy maker reported, students who already struggle with compulsory education were more likely to be advised by their parents not to opt for Informatics as an elective subject.

As to actual lessons, teachers who are better trained might be more likely to design their teaching and assessment to meet the needs of both their very able and their less able students. The question of what pedagogical approaches and topics cater best to the diverse needs of all students was not addressed within the scope of this case study. However, project and collaborative work together with personalized learning methods as practiced at the Case Study schools seem to cater adequately to the needs and interests of different students.

CS education can only be equitable and inclusive if some of its elements are compulsory for all students. However, making any subject compulsory is usually the result of a compromise. In most cases, only one or two hours of teaching a week is dedicated to a CS subject (which in itself does not even cover all CS-related concepts). So only minimal teaching of key CS components is ensured. Additional optional opportunities, including extra-curricular activities offered either by schools or grassroots organisations, continue to play a key role. This is particularly important for those students who have a high interest in the field, and the chance to go on and learn more may lead them to pursue a CS-related career path. For instance, high school students in Poland with an interest in CS can attend six additional hours of extended CS. These additional hours are mainly taken by students intending to follow a career related to CS.

Teacher recruitment and professional development – MCS2

A lack of qualified teachers[116] seems to be a key barrier to quality CS education in all three countries. Teachers who are not qualified are less likely to deliver high-quality teaching and assessment of CS. All countries report considerable differences in actual CS activities and the quality offered from school to school – although officially, CS education is (partly or fully) compulsory at the level of lower secondary education. Differences exist both in terms of qualified teachers and the schools' technical infrastructure. A lack of qualified teachers can also lead to issues when offering CS as an elective subject or extra-curricular activity for certain grades. In Croatia, the shortage of qualified teachers became particularly apparent when Informatics was introduced as an elective subject in the first four grades of all primary schools in 2020 and students showed high interest in the topic.

115. Computer Science in the Classroom Report - OKdo
116. 'Qualified teacher' refers to any teacher with the competence to teach CS - and help learners develop CT skills – in a sound and effective manner, irrespective of how that competence was acquired. It does not assume the teacher possessing official certification of that competence, or having followed a certified course of study to achieve it, although that may indeed be the case.

In other words, a sufficient number of qualified CS teachers is key to quality CS education. Countries do not have specific initiatives in place to recruit CS teachers. One issue is that those with the required knowledge in CS do not necessarily want to work as CS teachers. One reason put forward was that teaching salaries were not competitive. The Multiple-Case Study expert added that those teachers with deep knowledge and strong interest in CS often prefer a career where they can continue to develop this knowledge on a professional basis, rather than continuing with a teaching career.

Many countries across Europe start implementing their CS curricula without having a substantial fully-trained teacher workforce to draw from. This was particularly the case in UK-England and – to some extent – in Poland. One reason mentioned by the Multiple-Case Study expert is the "chicken and egg" problem: no one provides funding to train teachers for a subject or new curriculum that has not yet been put in place. In the three countries, some initial training opportunities were offered, and Croatia and Poland could even rely on qualified CS teachers. However, UK-England it was several years before large-scale teacher training opportunities came on stream. This meant that at times teachers were confronted with the task of teaching a new curriculum without being fully prepared or qualified.

In Croatia, large scale opportunities (funded by the European Social Fund as part of a Comprehensive Curricular Reform) were provided prior to the introduction of the curriculum. In preparation for the new curriculum, virtual classrooms on the Moodle platform opened in January 2018, which enabled both professional development and the sharing of experiences between teachers. In almost two years, more than 50,000 teachers participated in this training, which covered almost all teachers in Croatian primary and secondary schools.

Generally, communities of practice, school hub, and support from other teachers have proven to be key to complement more formal training opportunities. Especially in those first years of implementing new CS curricula, grassroots organisations often play a key role in providing training in the absence of opportunities, for example in UK-England. To successfully teach CS, teachers also need to be ready and willing to experiment, to be comfortable with knowing less than some of their students, and ready to fail and learn from it – something which can be a considerable challenge for many teachers.

One key question is how much training is enough to enable teachers to deliver a quality CS education. According to the Multiple-Case Study expert, for teachers to be able to teach CS in a way that fosters higher order thinking skills like problem-solving and logical thinking skills (that cannot easily be taught with other learning activities), they need to have at least a Bachelor in Informatics. For teachers to be able to implement simpler activities related to CT skills that motivate students and get them interested, the expert estimates that teachers need trainings of a few weeks. Obviously, his claim does not correspond to the reality of teacher training offered across Europe, that also include short term opportunities, online courses and self-study. Verifying this claim or not goes beyond the remit of this Multiple-Case Study. However, one question for further research could be: Does the training offered enable teachers to deliver CS education that actually fosters curricula goals, such as fostering problem-solving skills and other goals like bringing more students into IT related professions? One could hypothesise that CS education is likely to only defend its position as a separate and compulsory subject in the long run, if evidence can demonstrate that it actually caters to these goals. To that end, monitoring and evaluating current training offers and assessing further training needs is necessary.

Lessons learnt – MCS2

Main drivers

One of the main conclusions from the Multiple-Case Study is that a separate CS subject seems to be necessary to teach CS, covering proficient subject knowledge and the underlying foundations of the discipline. Teachers respond well to relatively open curricula that allow them autonomy in their implementation, provided this is combined with detailed guidance, teaching examples, and high-quality teaching examples. Using concrete teaching examples to create a shared understanding among teachers, policymakers and other stakeholders can be a way to address the lack of agreement on a common definition of CT.

Both in Croatia and Poland the reform of the CS curriculum was part of a more comprehensive curricular reform of all subjects, and this process seems to make it generally easier to create space for CT-related activities in the curriculum. However, that is not to say that CS concepts and related CT skills should only be taught as

part of a separate subject. On the contrary, skills acquired in CS education can indeed foster learning in other subjects. A key rationale for teaching CS concepts and related CT skills is that students need to be able to understand the digital world around them – both in their personal lives and professional careers. Any components of the CS curriculum related to this curriculum goal should be part of a compulsory subject to ensure that all students have the opportunity to acquire related knowledge, skills and attitudes. At the same time, elective courses and extra-curricular activities remain vital means allowing highly motivated students to excel, possibly going on to pursue a career in the field.

It is usually left to teachers to decide what tools they use and the Multiple-Case Study teachers tend to give their students choices. The starting point for teachers should be students' interests and needs rather than specific tools. In the context of relatively open CS curricula, pedagogical approaches like personalised and collaborative learning approaches can become powerful means to foster students' independent learning and problem-solving skills. The key to high-quality CS education is to have qualified and motivated teachers who are able and willing to experiment with new approaches and tools. This calls for support from colleagues, professional networks, Ministries of Education, and grassroots organisations. These last players can perform a crucial role, particularly in filling gaps in training provision during implementation of a new subject/curriculum, when a large-scale CS-confident teaching workforce needs to be mobilised rapidly, as was the case in UK-England.

Main Challenges

The lack of adequately prepared teachers is still a key challenge in the Multiple-Case Study countries. Teachers are, however, hard-working and able, as long as high-quality professional development and support is provided; indeed, they are not just able but positively willing to learn, as the one Case Study expert emphasised. Another challenge mentioned is that school leaders and senior leadership teams rarely make CS education a priority (e.g., in terms of timetable allocation, funding), as they have many other concerns to deal with.

Considerable differences exist between schools in what kind of CT-related activities they carry out (and how much time they spend on them), even where CS is a compulsory subject. Findings from the Multiple-Case Study suggest that policy makers sometimes lack a solid understanding of the core of CS education, knowledge that would enable them to define CS curricula in an optimal manner.

Policy recommendations

Pilots as enablers of new curricula

Smoother roll-out can be achieved by piloting a new curriculum with a limited number of schools to test guidelines, recommended pedagogical approaches and teaching resources. Such piloting took place both in Poland and Croatia.

Monitoring & research on the actual impact of CS curricula

For CS education to earn its place as a long-standing subject in education, more evidence is needed that the implementation of CS curricula actually leads to attainment of expected goals, such as development of problem-solving skills. So CS concepts and related CT skills need to be part of the national curriculum and of assessment and inspection processes. Ongoing and more systematic evidence-based monitoring and evaluation of curricula implementation is vital, as is further research.

Provide training opportunities at large scale that are fit for purpose

It is crucial to provide large-scale quality teacher training, exploiting synergies between Ministries of Education and other bodies like grassroots organisations. This training should foster a culture of safe mistake-making in the teacher community, so teachers have the courage to experiment and innovate. A key question for reflection is how much and what kind of training enables teachers to deliver CS-related education that actually leads towards the goals set out in CS curricula? To that end, more monitoring of teacher training is needed, together with further research.

Support schools in supporting each other

Hubs that connect different schools and help then to support each other can

reduce differences in the quality of CS education provision between schools. These have proved successful in UK-England, with 34 school-based hubs in place at the time of writing. The UK-England Case Study school provided an excellent example of such support: as part of remote teaching implemented in response to the COVID-19 pandemic, the Computing teacher connected both the school's own students and those from other schools in less privileged areas that lacked qualified teachers.

Provide sustained funding for infrastructure, training & research

Sustained funding is key to providing digital infrastructure and quality training to enable teachers to deliver high-quality inclusive education. Funding is also necessary to support more research on how to integrate CS concepts at school and teach CT skills to students. One question for research to address is how to refine contents for grade-level progression in CS curricula. Research should actively involve teachers and educators as research partners, as one Case study expert advocated.

Key findings from Multiple Case 3: CT *within other subjects* in lower secondary education

CT conceptualisation & understanding – MCS3

Although the actual term "Computational Thinking" is not generally used as such in the three countries in this Multiple-Case Study (MCS), the main understanding of CT in ISCED 2 (grades 7-9) revolves around three focal points: *programming*, *algorithm*, and *problem solving*.

In the three MCS3 cases, *programming* transcends the strict practice of computer coding. It is used in a more general sense to indicate a hands-on approach in which a set of practical skills and competences related to design and problem solving is fostered via the acquisition of competence in a programming language. Despite this terminological variation, the MCS interviews provided sufficient evidence that all three curricula at the lower secondary level encompass core concepts and practices that form part of what is generally understood as CT.

Although integrated within Maths & Technology subjects, algorithms and programming are conceptualised strongly in terms of digital competence in all three countries. In addition, in France, *algorithm* and *programming* are also considered in relation to Informatics.

In all three countries, programming is a supplementary topic added onto the existing Math and Technology curriculum, rather than fully integrated therein. During the interview with the MCS3 expert, it emerged that strong integration would involve detailing the connections between CT concepts and Maths & Technology concepts at the curriculum level (i.e., how certain CT concepts could improve the learning of Maths concepts). Moreover, the expert pointed out that a critical issue to be addressed is making mathematics more engaging, thus getting more learners involved.

CT implementation at school level – MSC3

In all three countries, core CT elements mainly fall within Maths and Technology. This approach accommodates both theoretical aspects concerning CT and Maths and more technical dimensions of computing (e.g., educational robotics) regarding CT and Technology.

In all three curricula, the amount of time to be allocated to programming is not specified in the curriculum itself but is left up to the individual teacher. What time allocation the teacher opts for depends on the overall time dedicated to the subject in which CT is embedded, and on the number of topics to be addressed.

In all three countries, during the three-year span in question (grades 7-9), **Maths** occupies the second largest space in the curriculum (after language), covering approximately 400 hours across the three years, specifically 418 hours in Finland, 378 hours in France, and 400 hours in Sweden.

The **Technology** subject occupies a comparatively shorter time: 76 in Finland, 162 hours in France, and 88 in Sweden.

In terms of curriculum topics/objectives to be covered, the situation varies

significantly across the three countries:

- In Finland, Algorithmic Thinking is one of the 20 objectives set for Maths;
- In the French curriculum, Algorithms & programming is one of the six content areas of the Maths curriculum;
- In Sweden, programming is included as a subtopic under Algebra and problem solving.

A similar situation applies concerning the topics/objectives to be covered in the Technology curricula of these three countries. This emerged not just from the interviews with experts but also from those conducted with teachers and in student focus groups held in all three countries; the perception emerged that only limited time could be dedicated to the teaching and learning of programming and algorithmics.

Another significant finding that emerged from the interviews with school leaders, teachers and students in this MCS regards provision of equipment and infrastructure (computers labs, digital devices, WIFI coverage, etc.) required to properly carry out learning activities in this area. This issue affects the three countries to different extents and has different degrees of impact.

The interviews with policymakers and experts highlighted the need to develop a policy strategy for keeping equipment and infrastructure up to date and fit for purpose in the medium to long term.

Pedagogical approaches, methods, and tools – MSC3

Overall, across the three cases, the theoretical part of programming (including the development of programming skills) is mainly addressed in Maths. At the same time, students also apply this knowledge in Technology when constructing computer-controlled artefacts, starting with the design and implementation of physical objects.

As to pedagogical approaches adopted in Mathematics, when teachers in France implement programming activities, they tend to favour collaborative learning approaches, whereas in the interviews conducted with Finnish students and teachers, pair-programming and individual-oriented learning approaches were described. In Technology, a project-based approach is adopted across the three countries in an effort to engage students in motivating activities.

In Finland, an active learning approach is generally encouraged, particularly in Maths, through the construction of simple programs. In programming activities, particular attention is placed on generalising solutions as an introduction to algorithmic thinking. In Technology-Craft, programming is mainly explored through the control of physical objects like robots.

In France, Maths teachers are encouraged to promote project work and foster collaboration among students in creative digital production of programs, applications, animations, etc. The use of the Scratch programming environment is recommended as very suitable for this pedagogical approach.

In Sweden, the Maths curriculum encourages students' use of digital tools and programming to investigate mathematical problems and concepts, make calculations, and interpret data. In lower secondary school, students are expected to gain a basic understanding of programming, dealing with concrete situations.

In the study of Technology, putting ideas into action is strongly connected to programming. In the interviews with Swedish teachers and student focus groups, activities were mentioned in which electronic equipment (e.g., alarms, auto or motion control lights) was connected to and controlled by programmable digital tools.

On the question of tools, it is worth noting that at ISCED 2 level, both the Finnish and Swedish interviewees referred to a shift from visual programming languages to text-based ones, like Python. This is in keeping with the increasing level of complexity in the topics studied and with the overarching objective to develop students' digital skills for future employment. By contrast, in France a visual-based programming language (namely Scratch) is still used at ISCED 2 level, following Ministry indications. The reasoning behind this approach is to continue with an easy-to-use programming environment, thereby reducing potential barriers for teachers and students.

CT assessment – MSC3

All three countries are developing strategies for summative assessment. Provisions have already been made in France and Sweden to include CT in the final national exam of Maths & Technology at the end of ISCED 2. This step has already been enacted in France (Diplôme national du brevet), while in Sweden the government is still in the process of integrating algorithm and programming elements in the final Maths exam.

In Finland, a mandatory final exam is not foreseen within the country's compulsory education system. However, criteria for summative assessment are provided in the curriculum. In 2021, these criteria were revised to provide a more homogenous and reliable basis for grading students' results at the end of lower secondary school.

As was made clear by the interviewees, the integration of programming in the final exam at the end of ISCED 2 is a strong indication of the importance attributed to this topic as part of the lower secondary curriculum.

Gender and equity issues – MSC3

While CT-related topics such as programming are typically targeted for explicit action to foster greater involvement of girls in computer-related areas, the interviews and focus groups conducted with representatives from the three MCS countries revealed a different perspective. Here, the fact that CT-related topics are addressed in compulsory subjects at ISCED 2 level is in itself considered a suitable approach for engaging both boys and girls. The same reasoning applies to social equity.

Teacher recruitment and professional development – MSC3

Since CT is integrated in all three countries' existing Maths and Technology subjects, there is no need to recruit new teachers in response to this curricular innovation. Instead, a significant challenge they all face is appropriate and timely teacher upskilling as, generally speaking, CT does not usually figure strongly in teachers' personal or professional background.

Following the introduction of programming in the Maths and Technology curricula, education authorities in all three countries have launched professional development initiatives to provide teachers with appropriate basic skills in programming. Three key aspects emerged from the MCS3 interviews on this matter: how to go about delivering programming courses for teachers; how to ensure satisfactory participation and engagement in such courses; what type of training opportunities are best suited to the targeted teaching population.

The National Agency for Education in Finland funds the development and implementation of targeted training courses through government funding of a regular series of competitive calls open to both public and private organisations and universities. This approach guarantees a wide variety and spread of training initiatives and opportunities that respond to specific trainee needs. Finnish teachers are required to fulfil an annual quota of 12-18 hours of professional development training. In this respect, two major challenges became evident from the MCS3 interviews. The first is that the quota is open to training in all subject areas, not just programming. The second is that a substitute class teacher needs to be found to fill in for any teacher attending a training event, thereby placing a strain on school resources and organisation, as not all schools can cover these demands easily.

In France, the Ministry of Education offered a two-day compulsory training course for ISCED 2 Maths and Technology teachers in 2015 around the outset of the country's curriculum reform at that time. In addition, a set of resources was developed, including guidelines, lesson plans, etc. Training courses are run by regional branches of the Ministry (Académie), as well as by universities and private sector organisations. It should be noted that teachers in France are not required to complete an annual quota of professional training.

In Sweden, the National Education Agency has offered online training courses and conferences about programming that not only targeted ISCED 2 Maths and Technology teachers, but all teachers in the system. In addition, several courses have been developed and implemented by universities. Although Swedish

teachers must fulfil an annual quota of 104 hours of professional development, this is regulated by agreements with trade unions, and teachers' participation in professional development initiatives is planned and managed by the school leader.

Overall, these teacher training courses generally reflect CT-related curricula contents. However, during the MCS3 interviews, teachers stressed the need to have access to more courses that help them develop their programming skills and to have access to inspirational ideas for integrating programming in their teaching practices.

Lessons learnt

Main drivers

- When integrating CT concepts within other subjects, it is crucial to have a clear policy (e.g., specifying CT topics/objectives to be covered in subject curricula) and a clear implementation strategy (e.g., including CT-related topics in the national final exam of the subjects involved).
- Integrating programming within Maths & Technology accommodates both theoretical aspects concerning CT and Maths and more technical dimensions of computing (e.g., educational robotics) regarding CT and Technology.
- When CT-related concepts are integrated within compulsory ISCED 2 subjects, they are more likely to reach all students, thereby addressing gender and equity issues.

.

Main Challenges

- Effectively integrating CT related skills within existing subjects requires a certain time span to permit a solid, well-grounded integration process, especially as this approach entails adequate upskilling of the teachers involved and enactment of effective new teaching practices.
- When integrated within different subjects, programming is often a supplementary topic added onto the existing Maths and Technology curriculum, posing teachers a difficult challenge to attain meaningful integration.
- Integration within other subjects places a burden on human resources (hours per week that teachers can devote to the teaching and learning of programming and algorithmics) and on physical resources (lab equipment, WIFI connections, etc.)

Policy recommendations

- Support the creation of a network that would foster collaboration among teachers, encouraging them to share experiences and concrete examples.
- Re-arrange and/or increase the number of hours dedicated to Maths and related subjects, possibly beyond the minimum requirements of the standard timetable.
- Provide more guidance so that a shared minimum level of teaching time is spent on programming and related concepts.
- Strengthen formalized synergies and links between Maths & Technology to facilitate the coordinated integration of programming in both these subjects.
- Encourage cooperation between teachers and researchers – with the help of educational decision-makers – to better understand suitable approaches and collect evidence on effective practices in integrating programming within Math & Tech.
- Monitor what works and what doesn't work, ensuring that feasibility and sustainability are considered.
- Plan effective policy strategies for maintaining adequate technology available in schools, possibly with the help of local organizations, community, and authorities.
- Create and encourage targeted in-service teacher training on programming and on how to handle the progression between grades in teaching programming according to age and using appropriate tools to that purpose.

Annex 5. Overview of core Computer Science contents in the nine case studies' curricula

Note: The following extracts have been drawn from the curricula of the nine countries selected for in-depth case study (see Annex 4). They capture the segments of those curricula that specifically concern basic CS contents and digital competence/digital literacy (text highlighted in light blue). CS Concepts present in the curricula listed below are marked in bold. In addition, these concepts are colour-coded in line with the coding adopted in fig. 11 (Section 5), i.e., **blue** for programming concepts, **orange** for those related to algorithms, and **green** for concepts concerning the relationship between algorithms and programming for developing a solution. CS concepts marked in bold **black** identify those addressed exclusively at lower secondary level.

MCS 1 - CT skills integrated as cross-curricular theme at primary level

CASE 1: LITHUANIA

Subject name:
Informatika

[EN:Informatics]

Informatics curriculum overview

The Informatics curriculum identifies six areas of achievement common to all grades 1 to 12, with specific achievements in each area.

- Digital content. Essential skills of working with digital devices; managing textual, graphical, numeric, visual, audial information; information visualization and presentation; digital content creation.
- Algorithms and programming. Solving problems: algorithm, action control commands (sequencing, branching, looping), programming in a visual programming environment for children.
- Problem solving. Essential technical and technological skills of working with digital devices: solving technical problems, evaluating and identifying suitable technologies for the selected problem, creative use of technologies.
- Data and information. Working with data skills: problem analysis, data collection, sorting, search and data management, content quality evaluation.
- Virtual communication. Social skills in a virtual environment: continuous learning, e-learning, communication via email, chats, social networks, sharing, collaboration, reflection.
- Safety and copyright. Digital safety, safe work with digital devices; ethics and copyright issues of information processing and usage; safety, ethics and copyright issues in virtual communication.

Educational goals in grade 3-4:

Area of achievement: Algorithms and programming

- **The concept of an algorithm** (e.g., creating a daily routine, a recipe, a **sequence** of steps).
- **Concept of program** (e.g., which commands are executed is important in an **algorithm** and in a program).
- **Conditional command**: conditional commands (e.g., **IF-THEN-ELSE or other**), with symbols or diagrams.
- **Repetition command (loop)**: how to solve a variety of problems involving sequences of commands, selection and repetition.
- **Representing an algorithm**: (e.g., to build or draw a geometric figure from a physical objects); describe the algorithm in words; write the algorithm in conventional signs or diagrams.
- **Decomposing the algorithm**: breaking down a problem into smaller parts; create programs, using **selection and repetition commands, sequences and logical operations**.
- **Checking the correctness of a solution**, **finding and correcting errors**: **sequencing of the steps** and presentation of the solution; discuss the strategy for solving the problem.

Area of achievement for data mining and information:

- Digital technologies in everyday life
- Representation of data (images) on a computer
- Patterns in data
- Representing data in diagrams
- Data and information security
- Data encryption

Area of achievement for (technological) problem solving:

- Use of digital devices
- Digital device impairments
- Selection of applications/apps
- Developing the technological skills necessary for learning

Lithuania Curriculum Portal: https://www.mokykla2030.lt/bp-projektai/

Informatics curriculum (ver. of 3/11/2021): https://www.emokykla.lt/bendrasis/bendrosios-programos/bendrojo-ugdymo-programu-projektai-2021-11-03

CASE 2: NORWAY

Mathematics

Core elements: Exploration and problem solving.

Exploration in mathematics is about the students looking for patterns, finding connections and discussing a common understanding. Students should place more emphasis on strategies and procedures than on solutions. Problem solving in mathematics is about the students developing a method for solving a problem they do not know before. Algorithmic thinking is important in the process of developing strategies and procedures for solving a problem and involves breaking down a problem into sub-problems that can be solved systematically. Furthermore, it means assessing whether sub-problems can best be solved with or without digital tools. Problem solving is also about analyzing and shaping known and unknown problems, solving them and assessing whether the solutions are valid.

Digital skills: Digital skills in mathematics mean being able to use graphing, spreadsheets, CAS, dynamic geometry software and **programming** to explore and solve mathematical problems.

Basic skills by end of grade 3:

- Explore equilibrium and balance in practical situations, represent this in different ways and turnover between the different representations.
- Create and follow rules and **step-by-step instructions** in games and related games the coordinate system.

Basic skills by end of grade 4:

- Explore and describe **structures** and **patterns** in play and games.
- Create **algorithms** and express them using **variables**, **conditions** and **loops**.

Science

Core elements: Students will understand, create and use technology, including **programming** and modeling, in work with science. By using and creating technology, students can combine experience and professional knowledge with thinking creatively and innovatively. Students must understand technological principles and working methods. They will assess how technology can contribute to solutions, but also create new challenges. Knowledge and competence in technology is therefore important from a sustainability perspective. Work with the core element technology must be combined with work related to the other core elements.

Digital skills: Digital skills in science are to be able to use digital tools to explore, register, calculate, visualize, **program**, model, document and publish data from experiments, fieldwork and other people's studies.

Basic skills by end of grade 4:

- Use tables and figures to organize data, create explanations based on data and present findings .
- **Compare models** with observations and conversation about why we use models in science.
- Explore technological systems composed of different parts, and describe how the parts work and work together.
- Design and make a product based on a requirements specification.

Arts and crafts

Digital skills: Digital skills in arts and crafts means being able to use digital tools and media for inspiration, testing, documentation, and presentation. It also involves using digital tools and **programming** in creative and creative processes. Knowledge of copyright and privacy policy when one uses one's own or others' images, films and creative work, is essential on all steps. The development of digital skills in arts and crafts goes from use simple digital tools and media, to shape your own digital products such as creates experiences and communicates feelings, ideas and opinions.

Basic skills by end of grade 4:

- Draw shape and depth using tools such as overlap and reduction.
- Apply simple compositional principles in photography and digital tools.
- Explore diversity in motifs and visual expressions in art from different continents and create a digital presentation.

Music

Digital skills: Digital skills in music are being able to use music technology to practice, create and experience music. This involves using digital tools creatively to do recording, processing and manipulating sound and using **programming** in creative work. Digital skills are also the exercise of digital judgment. It involves following copyright rules in the face of one's own and others' music and expel online ethics in interaction with others. The development of digital skills in music goes from using simple digital tools to shaping musical works, to use digital tools and technology strategically and varied to achieve appropriate and creative musical expressions. It also goes from being able to exercise privacy and netiquette in individual situations to be able to show good judgment and contribute to responsible collaboration in musical communities.

Source: Norwegian core curriculum https://sokeresultat.udir.no/finn-lareplan.html?fltypefiltermulti=Kunnskapsl%C3%B8ftet%202020

CASE 3: SLOVAKIA

Subject name:
Informatika

[EN:Informatics]

Characteristics of the Informatics subject

In the subject of Informatics, two components are intertwined. One component focuses on gaining specific experience and skills in working with computers and applications – working with digital technologies. The second component is aimed at building a foundation in computer science. Mainly on problem solving using computers. The first component forms the basis of informatics teaching in primary education and is largely interwoven throughout lower secondary education. The experience gained through practical work in this area is then a good prerequisite for mastering the second component, which dominates the teaching of computer science in secondary school. At the same time, however, the second component already appears in primary education, albeit in a very simple form. At the same time, informatics prepares pupils to make correct use of the skills and knowledge acquired in this way in other subjects.

Objectives (grades 3-4):

1. Consider information and various representations, use appropriate tools for their processing.
2. Think about **algorithmic**, search for and find **algorithmic solutions** to problems, **create instructions**, **programs** according to given rules.
3. Logically consider, argue, evaluate, make reasoned decisions.
4. Know the principles of software and hardware and use them in solving computer problems.
5. Communicate and collaborate through digital technologies, obtain information on the web.
6. Know how informatics has affected society.
7. Understand the risks on the Internet, are able to prevent them and solve problems that arise.
8. respect intellectual property.

Objective 2. *Think about algorithms, search for and find algorithmic solutions to problems, create instructions, programs according to given rules.*

Performance by end of grades 3-4:

Performance: *problem analysis*
- propose a solution, express a plan for a solution,
- decide on the truth/falsity of a statement (proposition),
- select elements or options according to the truth of the statement,
- consider different solutions.

Performance: *interactive solution building*
- solve the problem by direct control of the executor (e.g., robot, co-worker),
- apply elementary commands of a given language (from the command dictionary) to control the executor.

Performance: *using a **sequence** of commands*
- solve the problem by placing orders in **sequence**,
- add, complete, modify a solution in progress, interpret the **sequence** of commands,
- find an error in a **sequence** of commands.

Performance: *interpretation of the solution notation*
- implement instructions, procedure, **algorithm** for solving the problem – take it, step through the solution, simulate the activity of the executor.

Performance: ***finding, correcting errors***
- **find the error** in the result after the **algorithm** has been executed,
- **find and correct an error** in the instructions, in the solution notation,
- discuss their solutions.

Source: https://www.statpedu.sk/sk/svp/inovovany-statny-vzdelavaci-program/inovovany-svp-1.stupen-zs/matematika-praca-informaciami/

MCS 2 - CT skills as part of a distinct computing subject at lower secondary level

CASE 4: **CROATIA**

Subject name:
Informatika

[EN:Computer Science]

Characteristics of the Informatics subject

A special contribution of learning the Computer Science subject is reflected in the development of Computational Thinking, which also involves problem solving techniques:

- Information presentation by **abstractions**.
- **Logical reasoning** and data analysis.
- Solution automation by **algorithmic thinking**.
- Recognition, analysis and application of potential solution with the aim of achieving efficient results taking into account resources available.
- Formulating problems in a way that is appropriate for the use of computers and computer tools.
- **Generalising** the problem-solving process to be applicable to a whole series of similar problems.

There are four domains by which the goals of Computer Science will be realised:

- Computer Science is comprised of basic knowledge and concepts of computer science and understanding of digital presentation, storage and transfer of data by computer, digital devices or networks. These contents are studied in the domain of *Information and Digital Technology*.
- The domain of *Digital Literacy and Communication* provides basic digital competencies essential for a quality application of technology while completing daily duties.
- The domain *e-Society* is based on the fact that we live in an information society where digital technology impacts every aspect of life. Topics such as network security, data protection, cyberbullying and maintaining one's digital reputation.

It is necessary to develop **logical and algorithmic thinking** that is important to formulate problems in a way suitable for their solving by using computers, and it can be applied to other areas and everyday life. **Computational Thinking** is the fundamental approach in developing problem-solving ability and **programming** skills. The emphasis is on learning the process of creating an application from the concept to the final product and not purely on learning the syntax and semantics of a **programming** language. Activities and contents of outcomes in the domain of *Computational Thinking and Programming* develop innovation, creativity and entrepreneurial spirit and provide valuable knowledge that can be incorporated into future professional life.

Source: https://mzo.gov.hr/okvirni-godisnji-izvedbeni-kurikulumi-za-nastavnu-godinu-2020-2021/3929

Domain Computational Thinking and Programming , Grade 5

Domain: Computational Thinking and Programming

B.5.1 uses **program** tools to create a **program** in which he uses input and output values and **repetition**.

B.5.2 creates an **algorithm** for solving a simple task, checks if the **algorithm** is correct, **discovers and fixes errors**.

Domain Computational Thinking and Programming , Grade 6

Domain: Computational Thinking and Programming

B.6.1 creates, monitors and readjusts **programs** that contain branching and **conditional looping** structures and anticipates behaviour of simple **algorithms** that can be presented with a diagram, spoken language or **programming** language

B.6.2 considers and solves more complex problems by **separating them into a string of subproblems**.

CASE 5: POLAND

Subject name:
Informatyka

[EN:Computer Science]

Learning objectives of the Computer Science subject

- To understand, analyse and solve problems based on **logical** and **abstract thinking**, **algorithmic thinking** and ways of representing information.
- **Programming** and problem solving using a computer and other digital devices: arranging and **programming algorithms**, organizing, searching for and providing access to information, using computer applications.

- Using a computer, digital devices and computer networks, including knowledge of the principles of digital devices and computer networks, and performing calculations and programs.
- Developing social skills, such as communication and cooperation in a group, including in virtual environments, participation in team projects, and project management.
- To respect the law and safety rules. Respect the privacy of information and data protection, intellectual property rights, etiquette in communication and norms of social coexistence, assessing risks associated with technology and their consideration for the safety of themselves and others.

Grades 4-6:

- Understanding, analysing and solving problems: create and organize as a **sequence** (linear) or tree (non-linear) information; formulate and write in the form of **algorithms instructions**; distinguish basic steps in **algorithmic problem solving**.
- **Programming** and problem-solving using computer and other digital devices: design, create and write in a visual **programming** language; **test the program** on a computer for compliance with the adopted assumptions and correct if necessary (**debug**); prepare and present solutions to problems, use text and graphics editors; gather, organise and select the results of the work and the necessary resources on the computer or in virtual environments.

- Using the computer, digital devices and computer networks: use devices to record images, sounds and videos, to collect, organise and select personal resources; use the network to find needed information and educational resources, navigating between sites, as a communication medium, to work in a virtual environment.
- Developing social skills (equality in access to technology, collaboration, team problem solving).
- Observance of the law and safety rules (health and safety rules, digital risks, IPR).

Grades 7-9:

Understanding, analysing and solving problems: formulate a problem in the form of a specification (i.e. describes **data** and results) and distinguished steps in **algorithmic problem solving**; applies basic **algorithms** in solving problems (e.g., searching and ordering); represent **logical values**, numbers, signs and text in a computer; perform experiments with **algorithms**; present examples of applications of computer science in other fields.

- **Programming** and problem-solving using computer and other digital devices: **design, create and test programs** in the process of problem solving (**conditionals** and **iterative instructions**, **functions** and **variables**, and **array** designs); **design**, **develop and test** software to control a robot; use computer applications, prepare documents; save results and **data**, search for information on the web.

- Using the computer, digital devices and computer networks: diagram the structure and functioning of a computer network; use various devices to create digital resources; correctly use CS terminology.
- Developing social skills: take part in various forms of cooperation (e.g., programming in pairs or in a team); critically assess information; identify the range of IT competences necessary for different professions.
- Observance of the law and safety rules: describes ethical issues and act ethically; distinguish between types of licences.

Source: http://isap.sejm.gov.pl/isap.nsf/DocDetails.xsp?id=wdu20170000356

CASE 6:

UK-ENGLAND

Computing

Purpose of study

A high-quality computing education equips pupils to use computational thinking and creativity to understand and change the world. Computing has deep links with mathematics, science and design and technology, and provides insights into both natural and artificial systems. The core of computing is computer science, in which pupils are taught the principles of information and computation, how digital systems work and how to put this knowledge to use through programming. Building on this knowledge and understanding, pupils are equipped to use information technology to create programs, systems and a range of content. Computing also ensures that pupils become digitally literate – able to use, and express themselves and develop their ideas through, information and communication technology – at a level suitable for the future workplace and as active participants in a digital world.

Attainments target key stage 3 (end of grade 9)

- **Design**, use and **evaluate** computational **abstractions** that model the state and behaviour of real-world problems and physical systems
- 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.
- Use two 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**.
- Understand simple **Boolean logic** [for example, **AND, OR and NOT**] and some of its uses in circuits and **programming**; understand how numbers can be represented in binary, and be able to carry out simple operations on binary numbers [for example, binary addition, and conversion between binary and decimal].
- Understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems.
- Understand how instructions are stored and executed within a computer system; understand how data of various types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits.
- Undertake creative projects that involve selecting, using, and combining multiple applications, preferably across a range of devices, to achieve challenging goals, including collecting and analysing data and meeting the needs of known users.
- Create, re-use, revise and re-purpose digital artefacts for a given audience, with attention to trustworthiness, design and usability.
- 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.

Source: https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study

MCS 3- CT skills integrated within other subjects at lower secondary level

CASE 7: FRANCE

Mathematics

Cycle 4 (grades 7 - 9)

Theme E – *Algorithm and programming*

Knowledge:
- **Notion of algorithm** and **program**
- Notion of computer **variable**
- Triggering of an action by an **event**
- **Sequences** of instructions, **loops**, **conditional** instructions

Related Skills:
- Write, develop (**test**, **correct**) and execute a **program** in response to a given problem

(Further specification in the related Ministerial guidelines*):
- **decomposition**: analyse a complicated problem, break it down into sub-problems, sub-tasks;
- **pattern recognition**: recognize patterns, configurations, invariants, repetitions, highlight interactions;
- **generalization** and **abstraction**: identify logical sequences and translate them into conditional instructions, translate recurrent patterns into loops, design methods linked to objects that translate the expected behaviour;
- **algorithm design**: write modular solutions to a given problem, reuse already programmed algorithms, program instructions triggered by events, design algorithms running in parallel.

Technology

Cycle 4 (grades 7 - 9)

Topic – *Informatic and programming*

Sub-topic content: *Understanding how a digital network work*
- Components of a network, architecture of a local area network, means of connection of a computer medium
- Concept of protocol, organisation of **protocols in layers**, **routing algorithm**
- Internet
- Environmental impact related to data storage and flow and information networks

- Sub-topic content: *Write, develop (**test**, **correct**) and execute a **program***
- Notion of **algorithm** and **program**
- Notion of computer **variable**
- Triggering of an action by an **event**
- **Sequences** of instructions, **loops**, **conditional** instructions
- Signal form and transmission
- **Sensor**, **actuator**, **interface**

Sub-topic content: *Adopting ethical and responsible behaviour*
- Develop good practice in the use of communicating objects.
- Analyse the environmental impact of an object and its components.
- Analyse the life cycle of an object.

Sub-topic content: *Information, communication, citizenship*
- Society and technological developments: measuring the societal impact of objects and technical systems on society and the environment.

Source: French National Curriculum for cycle 4 (grades 7-9): https://eduscol.education.fr/document/621/download
*Ministerial guidelines: https://cache.media.eduscol.education.fr/file/Algorithmique_et_programmation/67/9/RA16_C4_MATH_algorithmique_et_programmation_N.D_551679.pdf

CASE 8: FINLAND

Mathematics

Grades 7 - 9

[Objective, O20] To guide the pupil to develop his or her **algorithmic thinking** and skills in applying mathematics and **programming** in problem-solving.

[Content, C1] The pupils deepen their **algorithmic thinking**. The pupils **programme** while learning good **programming** practices. They use their own or ready-made computer **programmes** as a part of learning mathematics.

Crafts

Grades 7 - 9

[Content, C3] Embedded systems are used in crafts, i.e., **programming** is applied in designing and producing.

Programming competence in grades 7-9

Programming as such is only described in the National core curriculum for basic education in terms of a few objectives and contents, mainly in mathematics and crafts (see above). Finland's "New Literacy Development" programme provides further guidelines on programming skills*, which are divided into three main areas: 1) *Computational thinking*, 2) *inquiry-based work and producing*, and 3) *programmed environments and operating in them*.

1. Computational Thinking

- ***Logical thinking*** *and processing of information*: The pupil processes information contained in different **generalisations**, uses different ways of marking and implements **logical operations** with different types of information.
- Solving and modelling of problems: The pupil analyses problems and evaluates their possible solutions on the basis of different criteria and visualises problems and solutions with the help of **generalisations** and diagrams.
- *Concepts and basic structures of* ***programming***: The pupil understands the role of an **algorithm** and is able to design a **programme** that appropriately takes advantage of the basic **programming structures** such as **sequential**, **repeated** and **conditional** actions.
- Practical skills: The pupil programmes **programs** in different environments, is familiar with the basics of one **text-based programming language** and can interpret a software code based on it.

2. Enquiry Based Work and Producing

- *Co-creation processes*: The pupil understands different roles in groups and different ways of cooperation and works reciprocally and participating actively in **programming** projects.
- *Creative production*: In cooperation with others, the pupil plans and implements as a process a solution in which some kind of development platform, different sensors and automation are used.
- The pupil designs and implements a game, simulation or application that solves some kind of problem related to subjects or real life.
- ***Programming*** *as a tool for learning*: The pupil is familiar with technological applications related to different subjects and explains their operating principles. The pupil uses **algorithmic thinking** and **programming** in problem-solving and exploring related to different subjects and projects and in producing and presenting information.

3. Programmed environments and operating in them

- **Programmed technology** in different areas of life: The pupil is familiar with the **logic** in the operation of **algorithms**, **automation** and robotics and their applications in different areas of life. The pupil reflects on the opportunities provided by **programmed technology** and its risks and ethical aspects.
- Impacts of programmed technology in everyday life: The pupil is able to tell how digital services are personalised and how advertising is targeted at users. The pupil reflects on the role of programming and data collected by digital services in social influencing and exerting influence in society.

Source: National core curriculum [EN]: https://www.ellibs.com/fi/book/9789521362590/national-core-curriculum-for-basic-education-2014
*Ministerial guidelines: https://uudetlukutaidot.fi/ohjelmointiosaaminen/

CASE 9: SWEDEN

Mathematics

Aims: to develop knowledge in using digital tools and **programming** to explore problems and mathematical concepts.

Grades 7 - 9

Core content: *Algebra*

- How **algorithms** can be created and used in **programming**.
- **Programming** in different **programming** environments.

Core Content: *Problem Solving*

- How **algorithms** can be created, tested and improved when **programming** for mathematical problem-solving.

Technology

Aims: to develop students' technical expertise and technical awareness so that they can orient themselves and act in a technologically intensive world; to deal with technical challenges in a conscious and innovative way.

Grades 7 - 9

Core Content: *Technological solution*

- Technical solutions that use electronics and how they can be **programmed**.

Core Content: *Working methods for developing technological solutions*

- Pupils' own constructions in which they apply control and regulations, including with the aid of **programming**.

Core Content: *Technology, man, society and the environment*

- The internet and other global technical systems. The benefits, risks and limitations of these systems.
- The relationship between technological development and scientific progress. How technology has enabled scientific discoveries to be made, and how science has made possible technological innovations. Security when using technology, for example storing and protecting data.
- How cultural attitudes towards technology have an impact on men's and women's choice of occupation and use of technology.

Civics

Aims: to understand the importance of digitalisation for social development and for personal integrity.

Grades 7 - 9

Core content: *Information and communication*

- The role of the media in disseminating information, forming public opinion, as a source of entertainment and to scrutinise society's power structures.
- Different kinds of media, their structure and context, for example social media, websites and daily newspapers.
- Different types of media, their structure and content, such as the different parts of a newspaper. How individuals and groups are portrayed, e.g., on the basis of gender and ethnicity, and how information in digital media can be controlled by underlying **programming**.
- Opportunities and risks associated with the internet and digital communication, and how to act responsibly when using digital and other media with reference to social, ethical and legal aspects.

Source: https://www.skolverket.se/download/18.31c292d516e7445866a218f/1576654682907/pdf3984.pdf

GETTING IN TOUCH WITH THE EU

In person

All over the European Union there are hundreds of Europe Direct information centres. You can find the address of the centre nearest you at: https://europa.eu/european-union/contact_en

On the phone or by mail

Europe Direct is a service that answers your questions about the European Union. You can contact this service:
- by freephone: 00 800 6 7 8 9 10 11 (certain operators may charge for these calls),
- at the following standard number: +32 22999696, or
- by electronic mail via https://europa.eu/european-union/contact_en

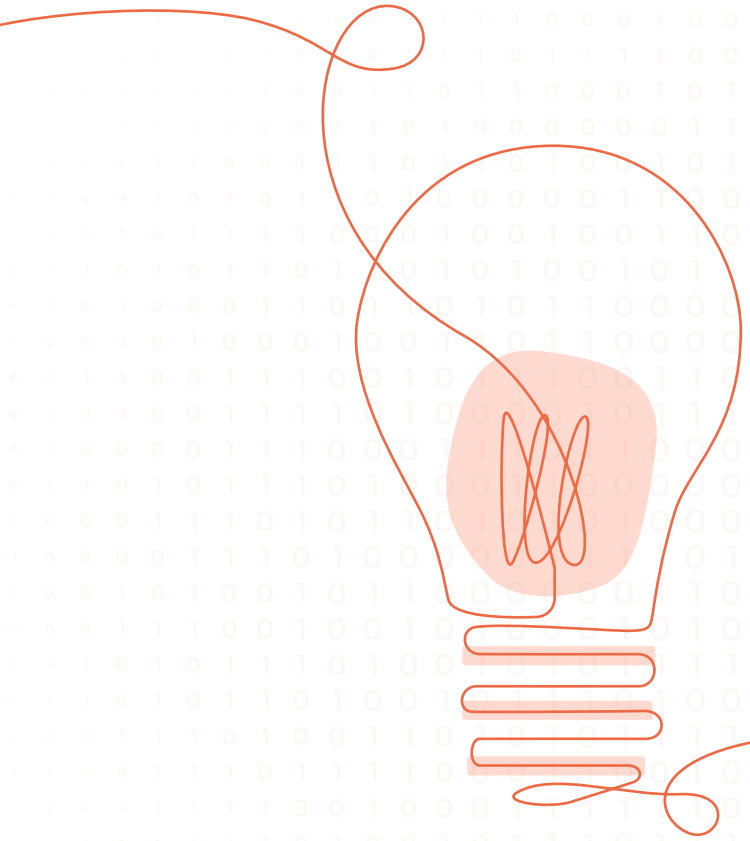
FINDING INFORMATION ABOUT THE EU

Information about the European Union in all the official languages of the EU is available on the Europa website at: https://europa.eu/european-union/index_en

EU publications

You can download or order free and priced EU publications from EU Bookshop at: https://publications.europa.eu/en/publications. Multiple copies of free publications may be obtained by contacting Europe Direct or your local information centre (see https://europa.eu/european-union/contact_en).

KJ-06-22-069-EN-N



The European Commission's science and knowledge service

Joint Research Centre

JRC Mission

As the science and knowledge service of the European Commission, the Joint Research Centre's mission is to support EU policies with independent evidence throughout the whole policy cycle.

EU Science Hub

ec.europa.eu/jrc

@EU_ScienceHub

EU Science Hub - Joint Research Centre

EU Science, Research and Innovation

EU Science Hub

doi:10.2760/126955
ISBN 978-92-76-47208-7

Publications Office
of the European Union