International Journal of Computer Science Education in Schools, April 2021, Vol. 4, No. 4
ISSN 2513-8359

International Trends in K–12 Computer Science Curricula through Comparative Analysis: Implications for the Primary Curricula

Michiyo Oda[1]
Yoko Noborimoto[2]
Tatsusya Horita[1]

[1]Tohoku University
[2]Tokyo Gakugei University

DOI: 10.21585/ijcses.v4i4.102

Abstract

The purpose of this study was to identify international trends in K–12 computer science curricula in countries that have introduced computer science education. Content analysis method was used to analyse the country-wide curricula of 10 countries which have introduced computer science education at the primary level. The K–12 Computer Science Framework was used as a theoretical frame to analyse the curricula. The results show that most countries begin their curricula with sub concepts of *algorithms*, *program development*, and under *impact of computing*, along with the practice of creating computational artifacts; then, countries expand upon computer science concepts and practices as learners progressed through the higher grades. Further, countries tend to introduce computer science concepts and practices in stages; once concepts and practices are introduced, they continue across multiple grades. Three approaches to implementing computer science education into the country-wide curriculum were found: introducing computer science (a) as an independent subject, (b) within multiple subjects, and/or (c) as a part of transversal competencies or an independent computer science curriculum with a cross-curricular approach. These study findings can contribute to a worldwide effort to introduce computer science education at the primary level.

Keywords: Computer science education; computer science curricula; primary school; computational thinking; cross-country analysis.

1. Introduction

1.1 Background

In this increasingly digital world, peoples' lives are heavily influenced by computing (French Academy of Sciences, 2013). This trend has affected people's lives, resulting in structural changes in society and triggering a growing need for a skilled science, technology, engineering, and math (STEM) workforce. For example, Fayer et al. (2017) reported that more than 800,000 STEM jobs were added to the U.S. economy between May 2009 and May 2015, and that STEM occupations grew by 10.5 percent during the same period, which is double the number of non-STEM occupations in the U.S. More importantly, for young people to thrive in this quickly changing, technology-based society, they must understand how computers work and be able to create innovative solutions while collaborating with others.

This societal change has stimulated concern regarding computer science education in K–12 education among European and American industries, academia, and policymakers. Published reports have detailed the need for

education reform, including integrating rigorous computer science education rather than information and communication technology (ICT) applications (Association for Computing Machinery, 2014; Computing at School Working Group, 2009; Gander et al., 2013; The Royal Society, 2012). Corresponding with these reports, curriculum reform initiatives have been introduced in various countries in K–12 settings. For example, European Schoolnet (2015) reported that 16 out of 21 European countries participating in their survey have already integrated computer science into their school curricula. Some countries have begun focusing their computer science education efforts within K–12 education, including the primary level. As of 2016, Bocconi et al. reported that policy initiatives of integrating computer science education in primary schools had been in place in several countries in Europe, Oceania, and Asia.

Computer science is a rigorous academic discipline on equal footing with other academic disciplines such as mathematics, physics, or geography, covering principles such as algorithms, data structures, programming, systems architecture, design, and problem solving (Comer et al., 1989; ECDL Foundation, 2015; The Royal Society, 2012). The idea that computer science is a discipline of its own is supported by the notions that computer science includes a body of knowledge that includes widely applicable ideas and concepts, theoretical frameworks to which these ideas and concepts apply, and fundamental concepts that do not change rapidly over time (The Royal Society, 2012). Based on the premise that computer science is an academic discipline, and that young learners should have the opportunity to learn concepts and principles of computer science, the Computing at School Working Group insists that there is a need for a consistent curriculum and body of knowledge on computer science education beginning at the primary level (Computing at School Working Group, 2009). Computer science is the common term in the U.S., while informatics is the common term in Europe (Gander et al., 2013). Although computer science, informatics, and computing have slightly different definitions, they all have been recognized as having a similar meaning (Heintz et al., 2016; Rolandsson & Skogh, 2014). Furthermore, computer science and programming (i.e. coding) are often recognized as equivalent in K–12 education (Association for Computing Machinery et al., 2016). In this study, computer science is used as an umbrella term in the general discussion. We chose the term used in the countries' curricula when discussing the specific countries.

In addition, programs for students enrolled in secondary school through university generally expand upon on the learning that occurs during primary programs. However, when considering computer science education, no successful models of curricula exist yet at the primary level. The introduction of computer science education to this age group is relatively new, as computer science education has been primarily set aside for higher education level learning (Heintz et al., 2016). Therefore, research on computer science curriculum at the primary level is a gap in the literature and must be further investigated. One way to address this issue is to explore international trends in K–12 computer science curricula from countries that have already integrated computer science into its national curricula. Through this exploration, meaningful suggestions may emerge and contribute to the development of primary level curricula.

1.2 Comparative Education Research on K–12 Computer Science Education

There are three approaches to consider in comparative education research on computer science education. The first approach is scrutiny and analysis of the educational system of computer science within one or more countries. For instance, in 2014 and 2015, ACM Transactions on Computing Education (TOCE) published special issues on computer science education in K–12 schools. These special issues included case studies on K–12 computer science education and showed great diversity in computer science education approaches adopted in each country based on each country's history and tradition of the educational system (Tenenberg & McCartney, 2014). In this special issue, Gal-Ezer & Stephenson (2014) compared K–12 computer science education in Israel and the U.S., focused on curriculum development, and reported common issues and challenges in areas in curriculum development and teacher training. Choi et al. (2015) reported current issues in introducing computer science education in K–12 education in Korea by analyzing education environments, including educational systems, curricula, and teaching environments. This first approach shows the detail of each country's computer education status but does not tell the trends of computer science education.

The second approach is comparative research through a cross-country comparative analysis. For example, Bocconi et al. (2016) reported a recent trend in computer science education in compulsory education among European countries through desk research, a survey of the Ministries of Education in 18 countries, and expert semi-structured interviews. One of their findings was that in 10 countries in Europe, including England, Poland, and Italy, computational thinking (i.e. computer science) is included in school curricula at the primary level.

Heintz et al. (2016) reviewed the introduction model of K–12 computer science education in 10 countries by analyzing the relevant documents for each country and found that the common model is to make computer science education compulsory in primary school and elective in secondary school. So et al. (2020) introduced computational thinking education (computer science education) status in the Asian Pacific Region by reviewing five empirical studies and one literature review study and showed the current research trends in the field of computer science education and teachers' perception in the region. The second approach presents computer science education trends in worldwide or regions; however, little research has been conducted in terms of curriculum trend.

The third approach is comparative research across multiple countries' reports. For instance, the Ministry of Education, Culture, Sports, Science and Technology (2015) in Japan investigated 23 countries that integrated computer science education into compulsory education from all over the world, including England, Estonia, Germany, Russia, Argentina, Taiwan, and South Africa. These countries were selected using the criteria of higher-ranking countries in PISA 2012 and TIMSS 2011; all countries had implemented computer science education in compulsory education. The ministry conducted desk research and expert interviews on computer science education and published a document combining the countries' status reports. The status of computer science's introduction in primary education was included as a part of each country's report. The third approach also demonstrates computer science education trends worldwide. This approach combines with the first and second approach. However, comparative research on computer science curriculum at the primary education level is still limited.

These three approaches of comparative education research focus primarily on comparing computer science education implementation status among countries and regions. To date, there are only a few studies conducting a cross-country comparative analysis on K–12 curriculum in terms of computer science. In addition, there are only a few studies focusing on primary computer science education in comparative education research. For these reasons, it is imperative that a cross-country comparative analysis be conducted on K–12 computer science curriculum, which will offer implications for primary computer science curriculum and promote consistency in curriculum development nationwide.

1.3 Research Purpose

This study targeted 10 countries that had already introduced computer science education in K-12. The purpose of the research is to show international trends in K–12 computer science curricula, and as a result, get meaningful findings to apply to the primary level curricula. Since the introduction of computer science education at the primary level is relatively new, this study may contribute to the countries' efforts to introduce computer science education at this level. This research is guided by the following research questions:

RQ1. What are the common approaches for computer science education in K–12 among the countries?

RQ2. What are the common trends on K–12 computer science curricula among the countries?

With various definitions of school levels across countries, this paper will use the term K–12 to refer to primary and secondary education. The term primary refers to primary school, which often includes elementary school (except in Sweden, where primary school refers to Years 1–3, middle school refers to Years 4–6, and secondary school refers to Years 7–9). The term secondary refers to both lower secondary and upper secondary; lower secondary includes middle school, junior high school, junior secondary school, lower secondary school, and secondary school; upper secondary refers to high school, senior high school, senior secondary school, upper secondary school, and secondary school.

1.4 Analysis perspective

To analyze the computer science curricula, understanding the perspective of a theoretically founded high-quality curriculum is critical. According to Cuban (1992), the definition of curriculum is "a series of planned events intended for students to learn particular knowledge, skills, and values and organized to be carried out by administrators and teachers" (p. 221). Curriculum is important since it gives direction to instruction in the educational system, and the accumulation of the learning experiences stimulates students' lives and eventually contributes to the quality of their lives (Schmidt et al., 1997).

Schmidt et al. (2005) believed that coherence and rigor are important characteristics that define a high-quality curriculum. According to the researchers, a curriculum is coherent when the sequence of topics and

performances appear based on the hierarchical nature of the subject discipline, and the depth of the discipline aligning to the coherence is defined as rigor. They examined the content standards analysis of the TIMSS's top six countries in mathematics and top four countries in science, and they reported that the coherence and rigor of these TIMSS top-achieving countries differed strikingly from the U.S. national standards in mathematics and science. The curricula of TIMSS's top-achieving countries shows a pattern in which new topics are gradually introduced and taught for a few grades, and then different topics are kept in the curriculum. However, various U.S. national standards in mathematics and science show that many more topics are introduced in the first grade and stay longer in each grade than those of the high-achieving countries' curriculum.

In alignment with coherence and rigor, Goodland and Su (1992) stated that organization of the curriculum is critical, which involves scope, continuity, sequence, and integration. Ediger (1995) and Maker (1986) noted that that the high-quality curriculum is designed with scope and sequence. Scope represents the breadth of the curriculum as a horizontal perspective, and continuity and sequence are organized vertically (Goodland & Su, 1992). The scope of the curriculum refers to what is taught, while the sequence is the order and depth of the content tying to the succession of human development (Ediger, 1995; Goodland & Su, 1992). Continuity is the organization of the curriculum through which students revisit the content, and integration aims to involve elements such as concepts, skills, and values in the curriculum (Goodland & Su, 1992). Since continuity is a part of sequence, and integration is a process of curriculum development, this study focuses on scope and sequence as a perspective for curriculum analysis to evaluate a high-quality computer science curriculum.

2. Method

Content analysis method was used to analyze the 10 countries which will be described in great detail in the Content Analysis section.

2.1 Selection of National Curricula Documents

First, we reviewed papers and reports that examined large-scale comparative research on K–12 computer science education status worldwide and within specific regions; the three papers/reports investigated were "Developing Computational Thinking in Compulsory Education" (Bocconi et al., 2016), "Computing our future" (European Schoolnet, 2015), and "Research on Programming Education in International Countries" (Ministry of Education, Culture, Sports, Science and Technology, 2015). Then we identified countries referenced in the papers and reports that matched the following criteria during our investigation (i.e., May 2019 to September 2019):

- Countries that offered computer science curricula or curricula that had integrated computer science in K–12.
- Countries that had implemented country-wide computer science curricula in schools. We excluded countries that had developed curricula at the regional level only, such as Spain, Germany, Belgium, and Switzerland, based on the research by Bocconi et al. (2016).

The reason for setting the criteria as such is to gain the implications for computer science curriculum at the primary level from the perspective of consistency of K–12 curricula; therefore, the countries that have integrated computer science education in K–12 were chosen for this study.

To determine whether the countries referenced in the paper or report matched the above criteria, we conducted additional research using website information, documents published by the ministries of education or relevant institutions, and peer-reviewed papers and reports that review national computer science education integration status. When there were unclear items related to educational systems and computer science education in each country, we asked for clarification from either ministry of education or national researchers who wrote papers to introduce their countries' introduction status of K–12 computer science education. As a result, 10 countries—Australia, England (United Kingdom), Finland, France, Hong Kong, Korea, New Zealand, Poland, Portugal, and Sweden—were selected.

2.2 Collection of National Curricula Documents

The national curricula collection was conducted between May and September 2019. After selecting the 10 countries, we identified the grades and subjects in which the countries introduced computer science education using the three papers and reports mentioned in section 2.1, website information, documents published by the ministries of education or relevant institutions, peer-reviewed papers and reports that reviewed national computer

science education integration status, and information from ministries of education and national researchers in the K–12 computer science education field. Then, the 10 countries' K–12 national curricula for the grades and subjects in which the countries introduced computer science education were collected from the website of ministries of education or equivalent institutions. The primary sources for the curricula appear in **Appendix A**. We collected curricula for both compulsory education and post-compulsory education (the first year of primary school through the last year of secondary school). Although most countries have introduced pre-primary education and higher education, and some countries have introduced pre-primary education as compulsory, this study did not include either one. In addition, this study investigated general high school curriculum and did not include other types of high schools, such as technical high schools and vocational schools. Moreover, because several countries have introduced computer science as an elective subject in secondary school, this study involved both compulsory and elective subjects for analysis.

In Sweden, computer science has been offered as a part of digital competence education, and digital competence has been integrated into various subjects in upper secondary school (Skolverket, 2017); therefore, no standalone compulsory courses in computer science education exist. We referred to the commentary by the Swedish National Agency for Education, entitled "Få syn på digitaliseringen på gymnasial nivå" (Skolverket, 2017) to understand the curricula in this country. This is a supplementary material for teachers providing background understanding on the role of digitization in the curriculum (Bocconi et al., 2018). Within this supplementary document, we identified all subjects that were related to digital competence and investigated the identified subjects' curriculum to assess whether the learning contents matched the K–12 Computer Science Framework definitions of the concepts and practices following the process shown in section 2.3. When there were learning contents in the supplemental material that matched the definitions, we decided that the subject was integrated computer science.

2.3 Selection of the Analysis Framework

To identify international trends in K–12 computer science curricula, we conducted a cross-country comparative curriculum analysis. To do so, a theoretical framework was necessary. In this research, the K–12 Computer Science Framework (Association for Computing Machinery et al., 2016) was utilized to analyze the 10 countries' national curricula. The framework outlines key concepts and practices that students should obtain by completing computer science education and was founded on profound research and practices; it was developed by the K–12 Computer Science Framework Steering Committee, which includes the Association for Computing Machinery, Code.org, Computer Science Teachers Association, Cyber Innovation Center, and National Math and Science. The framework is organized into five core concepts, which are divided into 17 subconcepts (**Appendix B**), and seven core practices, which are divided into 23 achievement levels by the end of Grade 12 (**Appendix C**).

The Association for Computing Machinery et al. (2016) recommended that the concepts and practices should be implemented into the school curriculum to provide a meaningful computer science experience for students. According to the organization, the structure of the framework was intended to align with the structure of widely accepted education frameworks, such as the framework for K–12 Science Education (National Research Council, 2012). According to the Association for Computing Machinery et al. (2016), the core concepts "represent major content areas in the field of computer science", and the practices "are the behaviors that computationally literate students use to fully engage with the core concepts of computer science" (p. 52). The concepts and practices in the framework hold main content areas and behaviors to implement the computer science curriculum for all students in K–12 education.

Although the framework was developed in the U.S., we utilized it to conduct a cross-country curriculum analysis because the framework (a) benchmarks several countries outside of the U.S., including the United Kingdom, Germany, Poland, and New Zealand, introducing a multi-country perspective; (b) is based on rigorous research and builds on some of the oldest K–12 computer science curriculum; and (c) represents a "baseline literacy for all students" without providing advanced contents to study (Association for Computing Machinery et al., p. 15).

2.4 Content Analysis

Content analysis method was employed in this study. According to Cohen, Manion, and Morrison (2007), the content analysis process includes defining words or sentences in texts, coding and categorizing them, and counting the words, codes, and categories. This study utilized the content analysis process that Cohen, Manion, and Morrison (2007) provide for curriculum analysis.

2.4.1 Analysis Contents Identification

First, we identified the chapters or units that described the knowledge and skills that are expected to be taught in each of the 10 countries' curricula. Madaus and Kellaghan (1992) explained that curriculum includes six major components: (a) context, (b) broad educational aims, (c) objectives of specific curricula or learning units, (d) curricular materials, (e) transactions and process, and (f) outcomes. In this study, we investigated representing the third component, objectives of specific curricula or learning units, which includes knowledge and skills to be taught that provides a basis for designing classroom instructional activities (Madaus & Kellaghan, 1992). A diverse range of chapter or unit titles were identified in the 10 countries' curricula related to computer science education objectives and learning units; these titles differed by country, subjects, and grades. For example, Australia utilizes the title "Sequence of content F-10" in the digital technologies subject, and England uses the title "subject content" in the computing subject. A list of the chapter or unit titles in curricula included in the content analysis appears in **Appendix D**. For example, we identified the "progress outcome" within the two technological areas of "computational thinking for digital technologies" and "designing and developing digital outcomes" as the above mentioned third component from the New Zealand, Technology curriculum.

2.4.2 Division of Sentences

Each curricula was composed of multiple sentences. We divided these multiple sentences, identified in section 2.4.1, into single sentences. For example, in the New Zealand curriculum exemplified in the section 2.4.1, multiple sentences were included in the "progress outcome" within the technological areas of "computational thinking for digital technologies" and "designing and developing digital outcomes". We divided the sentences which were included in the "progress outcome" into single sentences. An example of a divided single sentence was "In authentic contexts and taking account of end users, students use their decomposition skills to break down simple non-computerised tasks into precise, unambiguous, step-by-step instructions (algorithmic thinking)" (Progress outcome 1). When the texts were not written in English and we could not find an English version, Google Translate was utilized to translate the text from the original language to English to understand the meaning. Li et al. (2014) investigated the reliability of using Google Translate by comparing Google Translate with human translation, finding that Google English translation showed a high correlation with both human English translation and an original Chinese text. Although this research investigated Chinese-to-English translation, we thought this result could be applied to other languages. When there were unclear words in the curriculum that could not be translated using Google Translate, we asked researchers in the field of K–12 computer science education from the country so that we could understand the meaning.

2.4.3 Separation of Sentences

After dividing the multiple sentences from the curricula into single sentences, we divided the sentences into smaller clauses with identifiable meanings. For example, in the New Zealand curriculum exemplified in the section 2.4.2, the sentence, "In authentic contexts and taking account of end users, students use their decomposition skills to break down simple non-computerised tasks into precise, unambiguous, step-by-step instructions (algorithmic thinking)" (Progress outcome 1) was further divided into two clauses; "In authentic contexts and taking account of end users" and "students use their decomposition skills to break down simple non-computerised tasks into precise, unambiguous, step-by-step instructions (algorithmic thinking)".

2.4.4 Tagging of Units

After dividing each single curriculum sentence into one or more clauses, we tagged each clause with the concepts (5 core concepts and 17 subconcepts) and the practices (seven practices and 23 achievement levels of the practices) based on the definition from the K–12 Computer Science Framework. When the clauses were categorized across multiple concepts or practices, we tagged them all. When the clauses did not have an appropriate concept or practice to categorize, we did not tag them, and removed them from further analysis. For example, in the New Zealand curriculum example, "In authentic contexts and taking account of end users" was tagged with practice 1.2. The sentence "students use their decomposition skills to break down simple non-computerised tasks into precise, unambiguous, step-by-step instructions (algorithmic thinking)" was tagged with both *algorithms* and practice 3.2. The definition of practice 1.2 is "students should recognize that users of technology have different needs and preferences and that not everyone chooses to use, or is able to use, the same technology products" (Association for Computing Machinery et al., 2016, p. 74). The definition of *algorithms* is

"Algorithms are designed to be carried out by both humans and computers. In early grades, students learn about age-appropriate algorithms from the real world. As they progress, students learn about the development, combination, and decomposition of algorithms, as well as the evaluation of competing algorithms" (Association for Computing Machinery et al., 2016, p. 91). In addition, practice 3.2. represents decomposing real-world problems into manageable subproblems (Association for Computing Machinery et al., 2016). As shown in this example, we compared each clause to the definition of the concepts and practices from the K–12 Computer Science Framework. When the definition matched the meaning of the clauses, the clauses were tagged with the concepts and practices.

2.4.5 Review

One researcher categorized the clauses according to the concepts and practices, and the other researchers, who are subject specialists in computer science education, reviewed them. All researchers jointly discussed the categorization when there was a discrepancy to reach an agreement. We used this content analysis approach to generate country mapping tables by the concepts and practices (Tables 5 and 6).

3. Results

3.1 RQ1. What are the common approaches for computer science education in K–12 among the countries?

Table 1 summarizes the educational system and state of computer science education introduction by country. All countries revised their national curricula within a few years of the first implementation and introduced or reinforced computer science education. The country with the earliest development of a computer science K–12 curriculum was England in 2014, followed by Australia in 2015, Finland in 2016, and then France in 2016.

Table 2 displays the computer science integration curricula by country. Although the duration of compulsory education varies among countries, all of the countries introduced computer science as compulsory subjects during compulsory education. Some countries changed computer science from a compulsory subject to an elective subject as grade level increases. In addition, most of the countries in this study, except for Korea, integrated computer science curricula into a single subject, or multiple similar subjects, throughout compulsory education. Alternatively, Korea integrated computer science into "Practical Art" at the primary level, and then offers informatics at the secondary level.

Table 3 provides computer science integration subjects. "Technology" represents the subjects that include technology in the subject name, including Technology, Digital Technologies, General Study (Technology), Practical Arts (Technology/Home Economics), and Digital Science and Technology. "Independent computer science subject" stands for the independent computer science subjects, including Computing, Informatics, ICT, Digital and Computer Science, and Computer Applications B. Technology was the most introduced subject, followed by the independent computer science subject. Subjects of crafts, social sciences, language, and transversal competencies were found in only one country.

Table 4 demonstrates computer science introduction approaches. Several approaches for introducing computer science education were found. The first approach was introducing computer science as independent subjects (Australia, England, France, Hong Kong, Korea, New Zealand, Poland, and Portugal). The second approach was integrating computer science within multiple subjects (Finland, France, and Sweden). The third approach was introducing computer science as a part of transversal competencies or an independent computer science curriculum with a cross-curricular approach (Finland and Portugal). Some countries have introduced mixed approaches, differing by grade level. For example, Portugal introduced the third approach during lower primary level and transitioned to the first approach beginning in the upper primary level. In addition, some countries introduced different subjects within the same approach. For example, Portugal introduced computer science within ICT during grades 1–9 and changed the subject to Computer Applications B in grade 12.

Table 1. Educational System and State of Computer Science Education Introduction

Countries	Educational system and state of computer science education integration
Australia	In Australia, school education is compulsory between the ages of six and 16 (Year 1 to Year 9 or 10) (Australian Trade and Investment Commission, n.d.). Computer science education starts during the first year of school and continues through Year 10; there is no mandated national curriculum at the final stage of secondary school (Year 11 and Year 12) (Falkner et al., 2019). Current Foundation –Year 10 Australian Curriculum has been in place since **2015**, and the next revised version will be published by the start of 2022 (Australian Government, Department of Education Skills and Employment, 2020).
England	Compulsory education in England includes schooling for children age five through 16 (Years 1–11) (GOV.UK, n.d.). At the years 10 and 11, students can additionally take GCSE (Falkner et al., 2019). Computer science is integrated into Computing curriculum, which was replaced by ICT curriculum in **2014** (Computing at School, 2013).
Finland	In Finland, compulsory education is between the ages of seven and 15 (Grades 1–9) and compulsory pre-primary education starts at the age of six. In the revised curriculum for basic education, computer science is integrated in mathematics, crafts, and transversal competencies. This new curriculum was implemented for grades 1–6 in **2016**. For grades 7–9, the implementation of the revised curriculum was staged in **2017, 2018, and 2019** (National Board of Education, n.d.).
France	Education is compulsory in France from age six to 16 (Grades 1–11) (Gueudet et al., 2017). There are three types of schools in upper secondary level: general, technological, and vocational (Gueudet et al., 2017). This study used curricula under the general school level for analysis. The revised curriculum of the primary and the lower secondary schools (Grades 1–9) was implemented in September **2016** (Gueudet et al., 2017). For upper secondary school, the course "Digital Science and Technology" became compulsory for the second class of upper secondary school (Grade 10) in **2019**, and "Digital and Computer Science" has been provided as an elective in the first class of upper secondary school (Grade 11) (Ministry of National Education and Youth, 2018).
Hong Kong	In Hong Kong, compulsory education is between the ages of six and 15 (Primary 1–6, Secondary 1–3). The revised curricula were implemented in **2017** (The Curriculum Development Council, 2017a, 2017c, 2017b). Computer science education is integrated into technology education from Primary 1 to Secondary 6. Technology education is subsumed in the General Studies curriculum at the primary level. At the lower secondary level, schools are adopting a subject-based learning approach. Computer science is offered within the "Computer Literacy" subject to implement learning element modules of the technology education curriculum. At the upper secondary level, the elective subject "Information and Communication Technology (ICT)" has been offered as technology education (The Curriculum Development Council, 2017c).
Korea	In Korea, compulsory education is between ages six and 15 (Primary 1–6, Secondary 1–3) (National Center on Education and the Economy, n.d.). The revised elementary school curriculum for 5th and 6th graders was implemented in **2019** (Korean Ministry of Education, 2015). The previous curriculum included an ICT unit in "Practical Art" for 12 teaching hours, but the new curriculum includes computer science in "Practical Art" for more than 17 teaching hours (The Institute for Democracy, 2018). For lower and upper secondary schools, the introduction of the revised curriculum was staged in **2018, 2019, and 2020**, depending on grades (Korean Ministry of Education, 2015). The lower secondary school informatics course was an elective in previous curriculum, but is now a compulsory course (The Institute for Democracy, 2018). The new upper secondary informatics course is elective, consistent with the prior curriculum (The Institute for Democracy, 2018).
New Zealand	Compulsory education in New Zealand is between the ages of six and 16 (Years 1–10).

	The revised technology curriculum (Years 1–13) was published in **2017** and is expected to be fully implemented by the start of the 2020 school year (Ministry of Education, 2017). The new curriculum strengthened digital technologies, including computational thinking for digital technologies and designing and developing digital outcomes, as a part of the curriculum (New Zealand Ministry of Education, n.d.).
Poland	In Poland, compulsory education starts at the age of seven and lasts until the completion of Year 8 in primary school (Grades 1–8) (European Commission, 2019). The revised informatics curriculum was implemented in primary school (Grades 1–8) in **2017** (Polish Republic, 2017). Post-primary schools include 4–year general secondary school and 5–year technical secondary school. For secondary education (Grades 9–12, 4–year general secondary school, or Grades 9–13, 5–year technical secondary school), the revised curricula were applied in the **2019/2020** school year in the first grade (Education Development Center & Ministry of Education, 2019). Informatics is mandatory for one hour per week for three years; an elective extended informatics course is available for two to three hours per week for three years during secondary education in the new curriculum.
Portugal	In Portugal, compulsory education lasts for 12 years, starting at age six and ending at age 18 (European Commission, 2017a). Compulsory education includes basic education, which is between ages six and 15 (Grades 1–9), and secondary education, which is between ages 15 and 18 (Grades 10–12) (European Commission, 2017a). The revised ICT curriculum was implemented in basic education (Grades 1–9) and Computer Applications B (Grade 12) in secondary education in the **2018/2019** school year (Directorate-General for Education & Government of the Portuguese Republic, 2018).
Sweden	In Sweden, compulsory school is between the ages of seven and 16 (Years 1–9) (European Commission, 2017b). The revised curricula was implemented in basic education in **2017** and became compulsory in **2018** for all schools (Bocconi et al., 2018). Upper secondary school (Years 10–12) is optional. A revised version of the upper secondary school curriculum was published in 2018 for strengthening students' digital skills.

Table 2. Computer Science Integration Curricula

Countries	Compulsory Education (Starting in the first year of primary school)	Post-Compulsory Education (Ending in the last year of secondary school)
Australia	Digital Technologies (F–8), Digital Technologies (Grades 9–10) *	-
England	Computing (Years 1–11)	-
Finland	Mathematics (Grades 1–9), Crafts (Grades 3–9), Transversal competencies (ICT competences) (Grades 1–9)	-
France	Mathematics (Grades 1–10), Mathematics (Grade 11) *, Science and Technology (Grades 4–6), Technology (Grades 7–9), Digital Science and Technology (Grades 10), Digital and Computer Science (Grade 11) *	-
Hong Kong	General Study (Primary 1–6), Technology (Lower Secondary 1–3)	Technology (Upper Secondary1–3) *
Korea	Practical Arts (Technology/ Home Economics) (Primary 5, 6), Informatics (Lower secondary 1–3)	Informatics (Upper secondary 1–3) *

New Zealand	Technology (Years 1–10)	Technology (Years 11–13) *
Poland	Informatics (Grades 1–8)	Informatics (Grades 9–11), Informatics (Grades 9–11) *
Portugal	ICT (cross curricular in Grades 1–4, Grades 5–9), Computer Applications B (Grade 12) *	
Sweden	Mathematics (Grades 1–9), Technology (Grades 1–9), Social Studies (Grades 4–9)	Social Studies (Grades 10–12), Swedish / Swedish as a second language (Grades 10–12), Computers and ICT (Grades 10–12) *, Information and communication (Grades 10–12) *, Technology (Grades 10–12) *

Note. * Signifies an elective subject.

Table 3. Computer Science Integration Subjects

Approaches	AL	EN	FN	FR	HK	KR	NZ	PL	PR	SW
Technology	✓✻			✓	✓✻	✓	✓✻			✓
Independent computer science subjects		✓		✻		✓✻		✓✻	✓✻	✻
Mathematics			✓	✓✻						✓
Crafts			✓							
Social Studies										✓
Language										✓
Transversal competencies			✓							

Note. ✓ signifies a compulsory subject; ✻ signifies an elective subject. AL (Australia), EN (England), FN (Finland), FR (France), HK (Hong Kong), KR (Korea), NZ (New Zealand), PL (Poland), PR (Portugal), SW (Sweden).

Table 4. Computer Science Introduction Approach

Approaches	AL	EN	FN	FR	HK	KR	NZ	PL	PR	SW
Independent subjects	✓✻	✓		✻	✓✻	✓✻	✓✻	✓✻	✓✻	
Multiple subjects			✓	✓✻						✓✻
Transversal competencies or independent curriculum with a cross-curricular approach			✓						✓	

Note. ✓ signifies a compulsory subject; ✻ signifies an elective subject. AL (Australia), EN (England), FN (Finland), FR (France), HK (Hong Kong), KR (Korea), NZ (New Zealand), PL (Poland), PR (Portugal), SW (Sweden).

In summary, these results show that all 10 countries introduced or reinforced computer science education during compulsory education, as compulsory subjects, as a result of recent curriculum reform. It was common for countries to introduce computer science as a single subject or multiple similar subjects throughout compulsory education. The approaches that countries used to introduce computer science curriculum included (a) in independent subjects, (b) within multiple subjects, and (c) as a part of transversal competencies or an independent computer science curriculum with a cross-curricular approach. Subjects within the computer science

curriculum most often included technology, followed by independent computer science subjects and mathematics.

3.2 RQ2. What are the common trends on K–12 computer science curricula among the countries?

This study analysed 10 countries' computer science curricula for common trends in concepts and practices. Tables 5 and 6 offer a description of each country's inclusion of specific concepts and practices using circle pie charts. With respect to the right half of the circle, the top third of the semi-circle represents grades 1 and 2, the next third represents grades 3 and 4, the last third represents grades 5 and 6. On the left side of the circle, the bottom half of the semi-circle represents grades 7–9, and the next half represents grades 10 through 11, 12, or 13, depending on each country's educational system. In the circles, the colour black represents compulsory subjects, while grey represents elective subjects. When the computer science subjects existed within the curricula, but the concepts or practices did not match, the circle remained blank (white). If there were no computer science subjects, the circle was not drawn (without border). In addition, when a concept or practice was included in at least one or more grades within a specific fan shape that represents several grades, the fan shape was shaded black or grey. For example, if a concept or practice was included in grade 3 but not included in grade 4, the fan shape of grades 3 and 4 was filled in black or grey, even if the concept included in grade 3 and not included in grade 4. This rule was applied to other grades.

3.2.1 Concepts Analysis in Terms of Scope

As shown in Table 5, most countries covered most concepts. On the other hand, in Finland, the sub concepts of *computing systems* and *networks and the internet* were scarcely covered. In addition, several sub concepts, such as *variables*, *control*, and *modularity* in *algorithms and programming* were lacking across countries. Finland is the only country that introduced computer science within multiple subjects, excluding the subject of technology. Other countries that integrated computer science within multiple subjects included technology as one of subjects.

The results indicate that most countries in this study include almost all concepts from the K–12 Computer Science Framework in their computer science curricula. The countries that approached computer science integration with multiple subjects but did not include technology or independent computer science within the subjects (e.g., Finland), had a tendency for excluding *computing systems* and *networks and the internet* in their curricula.

3.2.2 Practice Analysis in Terms of Scope

Table 6 presents the description status in each country by practice. According to the Association for Computing Machinery et al. (2016), both the concepts and practices should be introduced to provide meaningful computer science experiences for students rather than only focusing on concepts alone. Some of the countries covered the concepts well, but lacked reference to practices 1 and 2. Practice 1 represents "considering the needs of diverse users during the design process is essential to producing inclusive computational products" (Association for Computing Machinery et al., 2016, p. 74) and practice 2 represents "the process of performing a computational task by working in pairs and on teams" (Association for Computing Machinery et al., 2016, p. 75). England lacked both practice 1 and 2, followed by Korea and Poland, which lacked practice 1, and Sweden, which lacked practice 2.

Practices 3 to 6 refer to delineated computational thinking in K–12 Computer Science Framework (Association for Computing Machinery et al., 2016). Computational thinking is commonly mentioned in relation to computer science. Computational thinking is defined as "the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent" (Wing, 2010, p. 1). Computational thinking is a thinking process to maximize the power of computing to solve problems effectively with the thinking process prior to the coding. According to Association for Computing Machinery et al. (2016), computational thinking is at the heart of the computer science practices. On the other hand, several papers argue that computational thinking is more than that. Denning (2017) compared traditional computational thinking with new computational thinking that are generated by Jeannette Wing' influential statement on computational thinking that was published in 2006 and the resulting discussions on computational thinking and argues that programming ability nurtures computational thinking in traditional computational thinking but learning certain concepts nurtures programming ability in new computational thinking. In line with the argument by Denning, Y. Li et al. (2020) insist that computational thinking should not be restricted in computer science; rather, it is pervasive in daily lives and occupations.

Australia, England, France, Hong Kong, New Zealand, Poland, and Portugal described practices 3 to 6 in their curricula across multiple grades. These countries have asserted computational thinking's importance in their publications by their respective ministries of education or equivalent organizations (Australian Curriculum, Assessment, and Reporting Authority, 2015; Baron et al., 2014; Department for Education, 2013a; Ministry of Education, 2017). Some countries lacked just one practice among the four practices. For example, Sweden lacked practice 3 and Korea lacked practice 6. Practice 3 represents recognizing and defining computational practices, while practice 6 stands for testing and refinement of computing artifacts. Adding these missing practices into curricula may reinforce nurturing computational thinking in these countries.

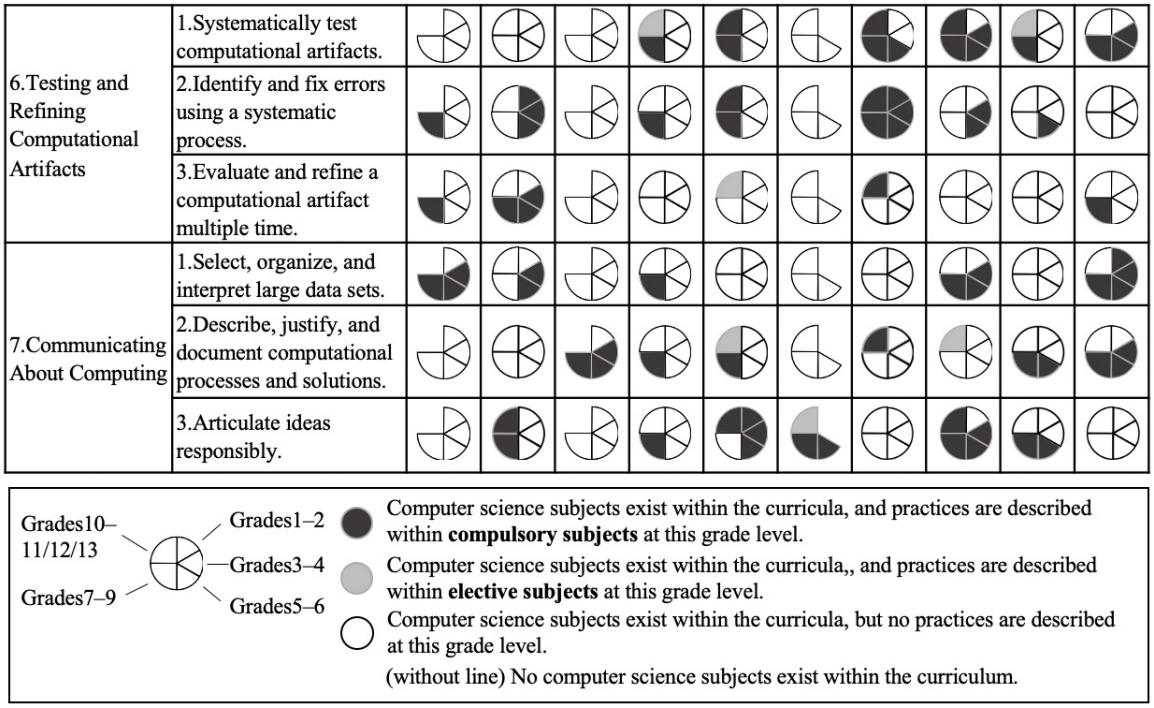
Table 5. Country Mapping by Concepts

Core Concepts	Subconcepts	AL	EN	FN	FR	HK	KR	NZ	PL	PR	SW
Computing Systems	Devices										
	Hardware and Software										
	Troubleshooting										
Networks and the Internet	Network Communication and Organization										
	Cybersecurity										
Data and Analysis	Collection										
	Storage										
	Visualization and Transformation										
	Inference and Models										
Algorithms and Programming	Algorithms										
	Variables										
	Control										
	Modularity										
	Program Development										
Impact of Computing	Culture										
	Social Interactions										
	Safety, Law, and Ethics										

Note. AL (Australia), EN (England), FN (Finland), FR (France), HK (Hong Kong), KR (Korea), NZ (New Zealand), PL (Poland), PR (Portugal), SW (Sweden). Australia: Compulsory education curriculum starts from Foundation, so this study includes Foundation in grades 1–2. The bottom half of the circle of the left side represents grades 7–10. New Zealand and Poland: The bottom half of the circle of the left side represents grades 7–8 and the next half represents grades 9–11/13.

Table 6. Country Mapping by Practices

Core Practices	By the end of Grade 12, students should be able to	AL	EN	FN	FR	HK	KR	NZ	PL	PR	SW
1.Fostering an Inclusive Computing Culture	1.Include the unique perspectives of others.										
	2.Address the needs of diverse end users.										
	3.Employ self- and peer-advocacy.										
2.Collaborating Around Computing	1.Cultivate working relationships.										
	2.Create team norms, expectations, and equitable workloads.										
	3.Solicit and incorporate feedback.										
	4.Evaluate and select technological tools.										
3.Recognizing and Defining Computational Problems	1.Identify complex, interdisciplinary, real-world problems.										
	2.Decompose complex real-world problems.										
	3.Evaluate whether it is appropriate and feasible.										
4.Developing and Using Abstractions	1.Extract common features.										
	2.Evaluate existing technological functionalities.										
	3.Create modules and develop points of interaction.										
	4.Model phenomena and processes and simulate systems.										
5.Creating Computational Artifacts	1.Plan the development of a computational artifact.										
	2.Create a computational artifact.										
	3.Modify an existing artifact.										



Note. AL (Australia), EN (England), FN (Finland), FR (France), HK (Hong Kong), KR (Korea), NZ (New Zealand), PL (Poland), PR (Portugal), SW (Sweden). Australia: Compulsory education curriculum starts from Foundation, so this study includes Foundation in grades 1–2. The bottom half of the circle of the left side represents grades 7–10. New Zealand and Poland: The bottom half of the circle of the left side represents grades 7–8 and the next half represents grades 9–11/13.

3.2.3 Concepts Analysis in Terms of Sequence

As can be seen in Table 5, common trends were found in the concepts presented in curricula across all 10 countries. Countries had similar tendencies in their timing of introducing certain concepts. For instance, *algorithms, programming,* and the sub concepts under *impact of computing* were described in curriculum in all countries, with descriptions starting in lower primary level and continuing throughout upper secondary level. Seven out of nine countries introduced *algorithms, program development,* and *safety, law, and ethics* at grades 1–2. These subjects continued throughout to the secondary level. Other sub concepts under *computing systems* and *networks and the internet* demonstrated a different trend from the *algorithms, programming,* and the sub concepts under *impact of computing*. The concepts had a tendency to be described from upper grades. For example, nine out of 10 countries described *hardware and software* and six out of 10 countries described *cybersecurity* at grades 7–9. Similarly, six out of 10 countries described *variables* at grades 7–9, and seven out of 10 countries described *control* at grades 5–6, which are the sub concepts under *algorithms and programming*. These sub concepts continued during multiple grades. It would appear that these sub concepts are to be learned in the higher-grade levels.

Within the concept of *data and analysis*, eight out of 10 countries included sub concepts of *collection* and all countries included *visualization and transformation*, although other sub concepts in the same core concept, such as *inference and models*, were included in four out of 10 countries. The reason for this result could be that *collection* and *visualization and transformation* were taught through the curriculum for mathematics in Finland, France, and Sweden; therefore, the sub concepts could have been reinforced by mathematics.

Troubleshooting was described the least across all 10 countries. Association for Computing Machinery et al. (2016) defined troubleshooting as follows: "When computing systems do not work as intended, troubleshooting strategies help people solve the problem" (p. 89). This sub concept addresses hardware troubles and solving problems through applying hardware and software knowledge. In this context, some sub concepts under *computing systems*, *networks and the internet*, and *algorithms* and *programming* may have supported learning of the concept *troubleshooting.* Therefore, the sub concept of *troubleshooting* requires higher-order thinking skills and various prerequisite knowledge, such as synthesizing and analysing the problems; therefore, this may be why

the number of *troubleshooting* descriptions was less common among the countries.

These results suggest common tendencies among the 10 countries regarding the sequence of computer science concepts presented across grade levels. For example, K–12 computer science curriculum usually starts with the concepts of *algorithms, program developmen*t, and the sub concepts under *impact of computing*, during the lower primary level and continues through to the upper primary level. Although *variables*, *control*, and *modality* are under the same core concepts of *algorithms and programming,* these sub concepts tend to be introduced in upper grades. Similarly, *computing systems* and *network and the internet* tend to be introduced in the upper grades.

Similarly, common trends across the 10 countries included each country described the sub concepts in stages and did not describe all concepts at the same grade. After the sub concepts described, the sub concepts continued during multiple grades. For example, French curriculum described sub concepts in stages such as *program development* in Cycle 2 (Grades 1–3), *algorithms* in Cycle 3 (Grades 4–6), and *variables* in Cycle 4 (Grades 7–9). After the sub concepts are described, these continued during multiple grades. This tendency noted in the French curriculum was similar to tendencies in the other nine countries' curricula. The results of this study are similar to results found by Schmidt et al. (2005). The researchers investigated curricula of the highest-achieving TIMSS countries in mathematics and science to identify elements of high-quality curriculum, finding patterns in how new topics are gradually introduced, continued for several grades, and then transition into different topics in the curriculum. Even though Schmidt et al.'s (2005) research investigated mathematics and science, the high-quality curriculum may show the same tendency. In the present study, we found that practices tended to be described only a few grades or multiple grades without consistency except for practice 5 and 6, unlike the sub concepts.

3.2.4 Practice Analysis in Terms of Sequence

As explained in the previous section, practices 3 to 6 were delineated as computational thinking in the K–12 Computer Science Framework (Association for Computing Machinery et al., 2016). As can be seen in Table 6, the number of countries that described practices 3, 4, and 6 tended to increase from lower secondary level to upper secondary level. On the other hand, practice 5, especially practice 5.2, was mentioned most commonly among all countries' curriculum across lower primary through upper secondary levels. The number of countries mentioning practice 5.1 in their curriculum increased from upper primary level to upper secondary level, and the same trend was seen with practice 6.1.

These results indicate that the introduction of computational thinking does not generally happen in order from practices 3 to 6; rather, it starts from creating computational artifacts as represented in practice 5.2 followed by the practices of planning the development of computational artifacts (practice 5.1) and testing computational artifacts (practice 6). Then moves to recognizing and defining computational problems (practice 3) and developing abstractions (practice 4) in the upper grades.

These results also suggest that computer science curricula tend to start not only with learning the concepts of *algorithms* and *program development*, but also with creating computational artifacts at the lower primary level. In addition, the introduction of computational thinking does not necessarily follow the order of practices 3 to 6; rather, it starts with creating computational artifacts (practice 5) and then expands into other practices gradually. Similar trends in timing of introduction were found between concepts and practices among the 10 countries.

4. Discussion

As the results showed, all 10 countries in this study have introduced or reinforced K–12 computer science education. The approaches that the 10 countries used to introduce computer science curriculum included either (a) as independent subjects, (b) within multiple subjects, and/or (c) as a part of transversal competencies or an independent computer science curriculum with a cross-curricular approach.

High-quality curricula require the existence of both scope and sequence. In addition, a high-quality computer science curricula should integrate both concepts and practices (Association for Computing Machinery et al., 2016); therefore, the scope and sequence of the curricula in this study are discussed in relation to concepts and practices in this section.

In terms of scope, most concepts in K–12 computer science curricula were described by nine countries; however, Finland, which integrated computer science within multiple subjects, but did not include the concept of technology or individual computer science as one of the subjects, also showing a lack in the sub concepts under

computing systems and *networks and the internet*. Since *computing systems* and *networks and the internet* were commonly found in technology or independent computer science subjects, introducing computer science without technology or independent computer science subjects could cause the lack of some concepts for students. However, this result does not mean that students in Finland do not learn the concepts because of following two reasons: (1) This study analysed the third component, objectives of specific curricula or learning units in curricula, as mentioned in the research methodology part, and does not include other parts of curricula, textbooks, or supplemental materials for the investigations, and (2) compared to other countries' curricula, Finland's curricula for the analysis included less information. In addition, Korea and New Zealand covered *networks and the internet* as an elective subject in upper secondary level. This means some students may not have a chance to learn *networks and the internet*; therefore, these countries could introduce *networks and the internet* as compulsory subjects to reinforce the concepts.

From the analysis of practices, practices 3 to 6 are delineated computational thinking in K–12 Computer Science Framework (Association for Computing Machinery et al., 2016). Seven countries among the 10 described practices 3 to 6 at one or some grade levels in their curricula. The three countries who did not reference practices 3 to 6 could further support students' computational thinking by adding these lacking practices to their curricula.

As for sequence, common trends among the 10 countries regarding the introductory grade levels of certain concepts and practices were identified. For example, *algorithms*, *program development,* and the sub concepts under *impact of computing*, were referenced in lower primary level through upper secondary level curricula. Sub concepts such as *hardware and software*, *cybersecurity*, *variables*, and *control* tended to be referenced in upper grades only. Practical activities, such as creating computational artifacts, were described in lower primary level through upper secondary level curricula. The results also suggest that nurturing students' computational thinking can be achieved by implementing practices 3 to 6, but not necessarily in the original order. Practice 5 is generally introduced in lower grades and then expanded upon through practices 3 and 6 in the upper grades.

The implications for computer science curricula at the primary level from this study are summarized as follows:

1) Computer science concepts start from *algorithms*, *program development*, and the sub concepts under *impact of computing* at lower primary level, then other concepts such as *computing systems* and *networks and the internet* are introduced in upper grades.

2) Computer science practices beginning with creating computational artifacts (practice 5) at lower secondary level, then expand to recognizing problems (practice 3), developing abstractions (practice 4), and testing and refining computational artifacts (practice 6) in upper grades.

3) Concepts and practices are introduced in stages, and after the concepts and practices are introduced, they continue across multiple grades.

4) The introduction of computational thinking does not happen in order from practices 3 to 6. It starts with practice 5 and expands into practices 3 and 6 in the upper grades.

Schools can translate the above implications into practice by first identifying an approach that fits best within their context, introducing computer science curricula as either (a) independent subjects, (b) within multiple subjects, and/or (c) as a part of transversal competencies or an independent computer science curriculum with a cross-curricular approach. Including technology or independent computer science subjects as one of subject offered is preferable so that the concepts of *computing systems* and *networks and the internet* are to be covered in the curriculum. Second, computer science education should be integrated within a single subject or similar multiple subjects throughout compulsory education. In this way, computer science curricula could remain consistent across grade levels.

5. Conclusion

This study analysed the curricula of 10 countries that have introduced computer science education beginning at the primary level with the goal of identifying trends in K–12 computer science curricula. These results offer meaningful findings for the development of computer science curricula at the primary level. A cross-country comparative curriculum analysis was completed, using the K–12 Computer Science Framework (Association for Computing Machinery et al., 2016) as the theoretical framework for interpretation. In this study, three approaches to implementing computer science were found: introducing computer science (a) as independent subjects, (b) within multiple subjects, and (c) as a part of transversal competencies or an independent computer science curriculum with a cross-curricular approach. We found common trends among the curricula of the 10 countries in how they implement the concepts and practices from the perspective of the scope and sequence.

Most countries begin their curricula at the lower primary level, covering *algorithms*, *program development*, and developing computational artifacts along with the sub concepts under *impact of computing*. Then, countries tend to gradually introduce other concepts and practices as grade level goes up.

Since the introduction of computer science education at the primary level is relatively new, and the most of research conducted in this field is still limited, this study contributes to the worldwide efforts to introduce computer science education at the primary level. In addition, this study also supports both countries who have curricula and those who do not have curricula yet. Computer science curricula has characteristics of both longevity and changeability; therefore, even countries that have already implemented computer science education could be encouraged to review the trends of other countries' curricula in order to improve the curricula. For example, England revised their ICT curriculum to a computing curriculum in 2014 and reviewed the introduction status in 2017, as published in a report by the Royal Society (2017). In Australia, the next revised version will be published by the start of 2022 (Australian Government, Department of Education Skills and Employment, 2020) so that Australia can include recent trends in computer science and lessons learned from the current curriculum.

5.1 Limitations

There are four primary limitations of this study. Firstly, this study targeted national curricula for analysis but did not investigate the curriculum or syllabus that teachers use in their classrooms. There are three types of curriculum: (1) the intended curriculum, which is the body of written content that policymakers expect to be taught, (2) the curriculum that is taught, including the informal and formal lessons in classroom, and (3) the curriculum that students learn (Cuban, 1992). Porter & Smithson (2001) defined the taught curricula (also called enacted curriculum) as "the actual curricular content that students engage in the classroom" (p.2). Similarly, Falkner et al. (2019) argued that the taught curricula includes the contents that are delivered within the classroom and adopted pedagogical approaches. Thus, the taught curricula have deeply connected to common teaching that teachers adopted in each subject and included critical aspects that relate to subjects. In this regard, Cuban (1992) remarked that there is a gap between what is intended and what is taught. Therefore, research on the intended curricula, what knowledge and skills teachers deliver in the classroom, and the outcome of the students as a result of what they learn is crucial and lacking the study on the taught curricula could cause one to miss important aspects of the subject. This study focused on only the intended curriculum because of the limitation of this study and did not include the taught curriculum, which can include researching textbooks or syllabi, and the learned curriculum, as evident through students' assessments.

Secondly, this study set the criterion for selecting countries that had implemented country-wide computer science curricula in schools. Because of the methodological limitations of this study, we set the criterion and tried to gain the implications for primary computer science curriculum from maturing national curricula within 10 countries. However, this criterion could risk eliminating the computer science curricula that have been implemented in countries with different governmental structures such as Cantons and regional governments, although they are more populous than some of the nations that this study included. In the future, we would like to expand criterion to these countries and areas to study their computer science education curricula.

Thirdly, this study targeted not all subjects within the curricula, but instead focused only on the subjects that were related to integrated computer science; all curricula were published by government or equivalent institutions. If we investigated all subjects within the 10 countries' curricula, including both computer science and unrelated subjects, the results may have changed.

Finally, Google Translate was utilized to translate some of the curriculum texts from the original language to English. Since this study targeted diverse countries, we thought that this translation method was the best way to achieve both feasibility and reliability based on the research by Li et al. (2014) that investigated the reliability of using Google Translate by comparing Google Translate with human translation, finding that Google English translation showed a high correlation with both human English translation and an original Chinese text. However, the further development of translation tools such as Google Translate may further contribute to the accuracy of obtaining information and promoting comparative curriculum research in the future.

Although this study identified trends in K–12 computer science curricula by conducting a cross-curricular analysis, the detailed analysis integrating the perspective of the educational system and history could contribute to the development of high-quality computer science curricula in primary school, since curriculum reflects each country's policy, history, and culture, future research could include the analysis of educational systems in terms of computer science curriculum, focusing on the few nations that have developed high- quality curricula among the 10 countries studied here.

References

Association for Computing Machinery. (2014). Rebooting the Pathway to Success: Preparing Students for Computing Workforce Needs in the United States. *ACM Pathways Report, Pathways.Acm.Org*. http://pathways.acm.org

Association for Computing Machinery, Code.org, Computer Science Teachers Association, Cyber Innovation Center, & Math and Science Initiative. (2016). *K–12 Computer Science Framework*. http://www.k12cs.org

Australian Curriculum, Assessment and Reporting Authority. (2015). *Digital Technologies: Sequence of content F-10*. Retrieved from https://www.australiancurriculum.edu.au/f-10-curriculum/technologies/digital-technologies/pdf-documents/

Australian Government, Department of Education Skills and Employment. (2020, June 12). *Australian Curriculum*. Retrieved from https://www.education.gov.au/australian-curriculum-0

Australian Trade and Investment Commission. (n.d.). *Australian education system*. Elcom Technology. Retrieved September 5, 2020, from https://www.studyinaustralia.gov.au/english/australian-education/education-system

Baron, G.-L., Drot-Delange, B., Grandbastien, M., & Tort, F. (2014). Computer Science Education in French Secondary Schools: Historical and Didactical Perspectives. *Trans. Comput. Educ.*, *14*(2), 11:1–11:27. https://doi.org/10.1145/2602486

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016). Developing Computational Thinking in Compulsory Education. Implications for policy and practice. *EUR - Scientific and Technical Research Reports*. https://doi.org/10.2791/792158

Bocconi, S., Chioccariello, A., & Earp, J. (2018). The Nordic approach to introducing computational thinking and programming in compulsory education. *Report prepared for the Nordic@BETT2018 Steering Group.* https://doi.org/10.17471/54007

Choi, J., An, S., & Lee, Y. (2015). Computing Education in Korea—Current Issues and Endeavors. *Trans. Comput. Educ.*, *15*(2), 8:1–8:22. https://doi.org/10.1145/2716311

Cohen, L., Manion, L., & Morrison, K. (2007). *Research Methods in Education, 6th Ed* (Vol. 1–6th Edition). Routledge.

Comer, D. E., Gries, D., Mulder, M. C., Tucker, A., Turner, A. J., & Young, P. R. (1989). Computing As a Discipline. *Commun. ACM*, *32*(1), 9–23. https://doi.org/10.1145/63238.63239

Computing at School. (2013). *Computing in the national curriculum A guide for primary teacher*. Retrieved from http://www.computingatschool.org.uk/data/uploads/CASPrimaryComputing.pdf

Computing at School Working Group. (2009).

Cuban, L. (1992). Curriculum Stability and Change. In P. W. Jackson (Ed.), *Handbook of Research on Curriculum: A Project of the American Educational Research Association* (pp.216-247).

Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM*, *60*(6), 33–39. https://doi.org/10.1145/2998438

Department for Education. (2013). *Computing programmes of study: Key stages 1 and 2 National curriculum in England*.

Directorate-General for Education, & Government of the Portuguese Republic. (2018). *Currículo Nacional—DL 55/2018* [National Curriculum—DL 55/2018]. Retrieved from https://www.dge.mec.pt/curriculo-nacional-dl-552018

ECDL Foundation. (2015). *Computing and Digital Literacy: Call for a holistic approach*. 9.

Ediger, M. (1995). Sequence and scope in the curriculum. *Education*, *116*(1), 159.

Education Development Center, & Ministry of Education. (2019, December 23). *Podstawa programowa kształcenia ogólnego dla liceum, technikum i branżowej szkoły II stopnia* [The core curriculum of general education for high schools, technical secondary schools and second-cycle industry schools]. Retrieved from https://www.ore.edu.pl/2018/03/podstawa-programowa-ksztalcenia-ogolnego-dla-liceum-technikum-i-branzowej-szkoly-ii-stopnia/

European Commission. (2017a, October 10). *Portugal Overview*. Eurydice - European Commission. Retrieved

from https://eacea.ec.europa.eu/national-policies/eurydice/content/portugal_en

European Commission. (2017b, October 10). *Sweden Overview*. Eurydice - European Commission. Retrieved from https://eacea.ec.europa.eu/national-policies/eurydice/content/sweden_en

European Commission. (2019). *Poland-Organisation of the Education System and of its Structure*. Eurydice - European Commission. Retrieved from https://eacea.ec.europa.eu/national-policies/eurydice/content/organisation-education-system-and-its-structure-56_en

European Schoolnet. (2015). *Computing our future: Computer programming and coding—Priorities, school curricula and initiatives across Europe*.

Falkner, K., Sentance, S., Vivian, R., Barksdale, S., Busuttil, L., Cole, E., Liebe, C., Maiorana, F., McGill, M. M., & Quille, K. (2019). An International Comparison of K-12 Computer Science Education Intended and Enacted Curricula. *Proceedings of the 19th Koli Calling International Conference on Computing Education Research*, 1–10. https://doi.org/10.1145/3364510.3364517

Fayer, S., Lacey, A., & Watson, A. (2017). *STEM Occupations: Past, Present, And Future*. U.S. BUREAU OF LABOR STATISTICS.

French Academy of Sciences. (2013). *Teaching computer science in France Tomorrow can't wait*.

Gal-Ezer, J., & Stephenson, C. (2014). A Tale of Two Countries: Successes and Challenges in K-12 Computer Science Education in Israel and the United States. *ACM Transactions on Computing Education*, *14*(2), 8:1–8:18. https://doi.org/10.1145/2602483

Gander, W., Antoine, P., Gérard, B., Barbara, D. G., Jan, V., Andrew, M., Avi, M., & Chris, S. (2013). *Informatics Education: Europe cannot afford to miss the boat*.

Goodland, J. I., & Su, Z. (1992). Organization of the Curriculum. In P. W. Jackson (Ed.), *Handbook of Research on Curriculum: A Project of the American Educational Research Association* (pp.327-344).

GOV.UK. (n.d.). *The national curriculum*. Retrieved August 22, 2020, from https://www.gov.uk/national-curriculum

Gueudet, G., Bueno-Ravel, L., Modeste, S., & Trouche, L. (2017). Curriculum in France: A National Frame in Transition. In D. Thompson, M. A. Huntley, & & C. Suurtamm (Eds.), *International Perspectives on Mathematics Curriculum* (pp. 41–70). International Age Publishing. https://hal.archives-ouvertes.fr/hal-01599059

Heintz, F., Mannila, L., & Farnqvist, T. (2016). A review of models for introducing computational thinking, computer science and computing in K-12 education. *2016 IEEE Frontiers in Education Conference (FIE)*, 1–9. https://doi.org/10.1109/FIE.2016.7757410

Korean Ministry of Education. (2015). *고등학교 교육과정(I,II,III)* [High School Curriculum (I, II, III)].

Li, H., Graesser, A. C., & Cai, Z. (2014). *Comparison of Google Translation with Human Translation*. 6.

Li, Y., Schoenfeld, A. H., diSessa, A. A., Graesser, A. C., Benson, L. C., English, L. D., & Duschl, R. A. (2020). Computational Thinking Is More about Thinking than Computing. *Journal for STEM Education Research*, *3*(1), 1–18. https://doi.org/10.1007/s41979-020-00030-2

Madaus, G. F., & Kellaghan, T. (1992). Curriculum Evaluation and Assessment. In P. W. Jackson (Ed.), *Handbook of Research on Curriculum: A Project of the American Educational Research Association* (p. pp.119-154).

Maker, C. J. (1986). *Developing Scope and Sequence in Curriculum*. 8.

Ministry of Education. (2017). *Technology in the New Zealand Curriculum*.

Ministry of Education, Culture, Sports, Science and Technology. (2015). *諸外国におけるプログラミング教育に関する調査研究* [Research on Programming Education in International Countries].

Ministry of National Education and Youth. (2018, November 13). *Numérique, technologie, sciences informatiques [Digital, technology, computer science]*. Retrieved from https://eduscol.education.fr/maths/actualites/actualites/article/numerique-technologie-sciences-informatiques.html

National Board of Education. (n.d.). *Perusopetuksen opetussuunnitelmien perusteet* [Fundamentals of basic

education curricula]. Opetushallitus. Retrieved January 26, 2020, from https://www.oph.fi/fi/koulutus-ja-tutkinnot/perusopetuksen-opetussuunnitelmien-perusteet

National Center on Education and the Economy. (n.d.). South Korea: Learning Systems. *NCEE*. Retrieved from http://ncee.org/what-we-do/center-on-international-education-benchmarking/top-performing-countries/south-korea-overview/south-korea-instructional-systems/

National Research Council. (2012). *A Framework for K-12 Science Education: Practices, Crosscutting Concepts, and Core Ideas*. https://doi.org/10.17226/13165

New Zealand Ministry of Education. (n.d.). *Technology in the NZC*. Retrieved January 29, 2020, from http://technology.tki.org.nz/Technology-in-the-NZC

POLISH REPUBLIC. (2017, February 24). *Rozporządzenie Ministra Edukacji Narodowej z dnia 14—Dziennik Ustaw* [Regulation of the Minister of National Education of 14—Journal of Laws]. http://dziennikustaw.gov.pl/du/2017/356

Rolandsson, L., & Skogh, I.-B. (2014). Programming in School: Look Back to Move Forward. *Trans. Comput. Educ.*, *14*(2), 12:1–12:25. https://doi.org/10.1145/2602487

Porter, A. C., & Smithson, J. L. (2001). Defining, Developing, and Using Curriculum Indicators. CPRE Research Report Series. https://eric.ed.gov/?id=ED477657

Schmidt, W. H., McKnight, C. C., Valverde, G., Houang, R. T., & Wiley, D. E. (1997). *Many Visions, Many Aims: A Cross-National Investigation of Curricular Intentions in School Mathematics*. Springer Science & Business Media.

Schmidt, William H., Wang, H. C., & McKnight, C. C. (2005). Curriculum coherence: An examination of US mathematics and science content standards from an international perspective. *Journal of Curriculum Studies*, *37*(5), 525–559. https://doi.org/10.1080/0022027042000294682

Skolverket. (2017). *Få syn på digitaliseringen på gymnasial nivå* [Get an idea of digitalisation at upper secondary level]. Retrieved from https://www.skolverket.se/publikationsserier/kommentarmaterial/2017/fa-syn-pa-digitaliseringen-pa-gymnasial-niva?id=3784

So, H.-J., Jong, M. S.-Y., & Liu, C.-C. (2020). Computational Thinking Education in the Asian Pacific Region. *The Asia-Pacific Education Researcher*, *29*(1), 1–8. https://doi.org/10.1007/s40299-019-00494-w

Tenenberg, J., & McCartney, R. (2014). Editorial: Computing Education in (K-12) Schools from a Cross-National Perspective. *ACM Trans. Comput. Educ.*, *14*(2), 6:1–6:3. https://doi.org/10.1145/2602481

The Curriculum Development Council. (2017a). *General Studies Curriculum Guide for Primary Schools (Primary 1 – Primary 6)*.

The Curriculum Development Council. (2017b). *Mathematics Education Key Learning Area Curriculum Guide (Primary 1 – Secondary 6)*.

The Curriculum Development Council. (2017c). *Technology Education Key Learning Area Curriculum Guide (Primary 1 – Secondary 6)*.

The Institute for Democracy. (2018). *소프트웨어 교육 현황과 개선 방향* [Software education status and improvement direction].

The Royal Society. (2012). *Shut down or restart? The way forward for computing in UK schools*. https://royalsociety.org/topics-policy/projects/computing-in-schools/report/

The Royal Society. (2017). *After the reboot: Computing education in UK schools*. https://royalsociety.org/topics-policy/projects/computing-education/

Wing, J. (2010). Computational Thinking: What and Why? *Unpublished Manuscript in Progress.* Referenced in http://www.cs.cmu.edu/~CompThink/papers/TheLinkWing.pdf.

Appendix A

Primary Sources of the National Curricula in This Study

Country	Curricula
Australia	Digital Technologies: Sequence of content F-10 (Australian Curriculum, Assessment and Reporting Authority, 2015)
England (United Kingdom)	Computing programmes of study: key stages 1 and 2 National curriculum in England (Department for Education, 2013) Computing programmes of study: key stages 3 and 4 National curriculum in England (Department for Education, 2013)
Finland	Perusopetuksen Opetussuunnitelman Perusteet 2014 [Basic Education Basics of the Curriculum 2014] (National Board of Education, 2014) Lukion Opetussuunnitelman Perusteet 2015 [Upper School Basics of the Curriculum 2015] (National Board of Education, 2015)
France	Programme du cycle 2 En vigueur à compter de la rentrée de l'année scolaire 2018-2019 [Cycle 2 Program Effective as of the beginning of the 2018-2019 school year] (Ministry of National Education and Youth, 2018) Programme du cycle 3 En vigueur à compter de la rentrée de l'année scolaire 2018-2019 [Cycle 3 Program Effective as of the beginning of the 2018-2019 school year] (Ministry of National Education and Youth, 2018) Programme du cycle 4 En vigueur à compter de la rentrée de l'année scolaire 2018-2019 [Cycle 4 Program Effective as of the beginning of the 2018-2019 school year] (Ministry of National Education and Youth, 2018) Sciences numériques et technologie Classe de seconde, enseignement commun [Digital Sciences and Technology Second Class, Common Teaching] (The Higher Program Council (CSP), 2018) Mathématiques Classe de seconde, enseignement commun [Mathematics Second Class, Common Teaching] (The Higher Program Council (CSP), 2018) Numérique et sciences informatiques Classe de première, enseignement de spécialité, voie générale [Digital and Computer Sciences First Class, Specialty Education, General Track] (The Higher Program Council (CSP), 2018) Mathématiques Classe de première, enseignement de spécialité [Mathematics First Class, Specialty Education] (The Higher Program Council (CSP), 2018)
Hong Kong	General Studies Curriculum Guide for Primary Schools (Primary 1 – Primary 6) (The Curriculum Development Council, 2017) Technology Education Key Learning Area Curriculum Guide (Primary 1 – Secondary 6) (The Curriculum Development Council, 2017)
Korea	초등학교 교육과정 교육부 고시 제 2015-74 호 [별책 2] [Elementary School Curriculum and Education Notice No. 2015-74] (Ministry of Education, 2015) 중학교 교육과정 교육부 고시 제 2015-74 호 [별책 3] [Middle School Curriculum and Education Notice No. 2015-74] (Ministry of Education, 2015) 고등학교 교육과정(I,II,III) [High School Curriculum (I, II, III)]. (Ministry of Education, 2015)
New Zealand	Technology in the New Zealand Curriculum (Ministry of Education, 2017)
Poland	Podstawa programowa kształcenia ogólnego z komentarzem. Szkoła podstawowa Informatyka [General Education Core Curriculum with Commentary. Elementary School Computer Science] (Education Development Center, and Ministry of Education, 2017) Podstawa programowa kształcenia ogólnego z komentarzem. Szkoła

	ponadpodstawowa: liceum ogólnokształcące, technikum oraz branżowa szkoła I i II stopnia Informatyka [General Education Core Curriculum with Commentary. Secondary School: General High School, Technical High School and Industry First- and Second-Degree Computer Science] (Ministry of Education, 2019)
Portugal	1.º CICLO DO ENSINO BÁSICO ORIENTAÇÕES CURRICULARES PARA AS TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO [1st Basic Education Cycle Curriculum Guidelines for Information and Communication Technologies] (Directorate-General for Education, and Government of the Portuguese Republic, 2018) 5.º ANO | 2.º CICLO DO ENSINO BÁSICO TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO [5th Year | 2nd Basic Education Cycle Information and Communication Technologies] (Directorate-General for Education, and Government of the Portuguese Republic, 2018) 6.º ANO | 2.º CICLO DO ENSINO BÁSICO TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO [6th Year | 2nd Basic Education Cycle Information and Communication Technologies] (Directorate-General for Education, and Government of the Portuguese Republic, 2018) 7.º ANO | 3.º CICLO DO ENSINO BÁSICO TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO [7th Year | 3rd Basic Education Cycle Information and Communication Technologies] (Directorate-General for Education, and Government of the Portuguese Republic, 2018) 8.º ANO | 3.º CICLO DO ENSINO BÁSICO TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO [8th Year | 3rd Basic Education Cycle Information and Communication Technologies] (Directorate-General for Education, and Government of the Portuguese Republic, 2018) 9.º ANO | 3.º CICLO DO ENSINO BÁSICO TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO [9th Year | 3rd Basic Education Cycle Information and Communication Technologies] (Directorate-General for Education, and Government of the Portuguese Republic, 2019) 12.º ANO | ENSINO SECUNDÁRIO APLICAÇÕES INFORMÁTICAS B [12th Year | Secondary Education Computer Applications B] (Directorate-General for Education, and Government of the Portuguese Republic, 2018)
Sweden	Läroplan för grundskolan, förskoleklassen och fritidshemmet [Curriculum for elementary school, Preschool Class and Kindergarten] (National Agency for Education (Skolverket), 2019) (English version exists.) Swedish, Swedish as a second language, Social studies, Technology, Information and communication, Computers and ICT (National Agency for Education (Skolverket), n.d., Subject plans in upper secondary school in English)

Appendix B

Concepts of K–12 Computer Science Framework

Core Concepts	Sub Concepts
Computing Systems	Devices
	Hardware and Software
	Troubleshooting
Networks and the Internet	Network Communication and Organization
	Cybersecurity
Data and Analysis	Collection

	Storage
	Visualization and Transformation
	Inference and Models
Algorithms and Programming	Algorithms
	Variables
	Control
	Modularity
	Program Development
Impact of Computing	Culture
	Social Interactions
	Safety, Law, and Ethics

Appendix C

Practices of K–12 Computer Science Framework

Practices	By the end of Grade 12, students should be able to
1. Fostering an Inclusive Computing Culture	1. Include the unique perspectives of others and reflect on one's own perspectives when designing and developing computational products.
	2. Address the needs of diverse end users during the design process to produce artifacts with broad accessibility and usability.
	3. Employ self- and peer-advocacy to address bias in interactions, product design, and development methods.
2. Collaborating Around Computing	1. Cultivate working relationships with individuals possessing diverse perspectives, skills, and personalities.
	2. Create team norms, expectations, and equitable workloads to increase efficiency and effectiveness.
	3. Solicit and incorporate feedback from, and provide constructive feedback to, team members and other stakeholders.
	4. Evaluate and select technological tools that can be used to collaborate on a project.
3. Recognizing and Defining Computational Problems	1. Identify complex, interdisciplinary, real-world problems that can be solved computationally.
	2. Decompose complex real-world problems into manageable subproblems that could integrate existing solutions or procedures.
	3. Evaluate whether it is appropriate and feasible to solve a problem computationally.
4. Developing and Using Abstractions	1. Extract common features from a set of interrelated processes or complex phenomena.
	2. Evaluate existing technological functionalities and incorporate them into new designs.
	3. Create modules and develop points of interaction that can apply to multiple situations and reduce complexity.
	4. Model phenomena and processes and simulate systems to understand and evaluate potential outcomes.
5. Creating Computational Artifacts	1. Plan the development of a computational artifact using an iterative process that includes reflection on and modification of the plan, taking into account key features, time and resource constraints, and user expectations.

	2. Create a computational artifact for practical intent, personal expression, or to address a societal issue.
	3. Modify an existing artifact to improve or customize it.
6. Testing and Refining Computational Artifacts	1. Systematically test computational artifacts by considering all scenarios and using test cases.
	2. Identify and fix errors using a systematic process.
	3. Evaluate and refine a computational artifact multiple times to enhance its performance, reliability, usability, and accessibility
7. Communicating About Computing	1. Select, organize, and interpret large data sets from multiple sources to support a claim.
	2. Describe, justify, and document computational processes and solutions using appropriate terminology consistent with the intended audience and purpose.
	3. Articulate ideas responsibly by observing intellectual property rights and giving appropriate attribution.

Appendix D

Analysis Contents from the National Curricula in This Study

Country	Contents
Australia	Sequence of content F-10 Strand: Knowledge and understanding (Digital Technologies, F–10)
England (United Kingdom)	Subject content (Computing, Years 1–11)
Finland	Matematiikan tavoitteisiin liittyvät keskeiset sisältöalueet [Key content areas related to mathematics objectives] (Mathematics, Grades 1–9) Käsityön tavoitteisiin liittyvät keskeiset sisältöalueet [Key content areas related to craft objectives] (Crafts, Grades 3–9) Laaja-alainen osaaminen [Extensive expertise] (Transversal competencies (ICT competences), Grades 1–9)
France	"Nombres et calculs [Numbers and calculations]" and "Espace et géométrie [Space and geometry]" (Mathematics, Grades 1–6), Connaissances, Compétences associées [Knowledge, Associated Skills] (Mathematics, Grades 7–10), Histoire des mathématiques [History of Mathematics] and Capacités associées [Associated Capabilities] in "Algorithmique et programmation [Algorithms and programming]", (Mathematics, Grades 11) Connaissances et compétences associées [Associated knowledge and skills] (Science and Technology, Grades 4–6) Connaissances et compétences associées [Associated knowledge and skills] (Technology, Grades 7–9) Contenus, Capacités attendues [Content, Expected Capacities] (Digital Science and Technology, Grades 10) Contenus, Capacités attendues [Content, Expected Capacities] (Digital and Computer Science, Grade 11)
Hong Kong	Knowledge and Understanding, Skills, Values and Attitudes in "Strand 3 (Science and Technology in Everyday Life)" and "Strand 6 (Global Understanding and the Information Era)", (General Study, Primary 1–6) Knowledge contexts (Information and Communication Technology (ICT)), Modules (K1 Computer Systems, K2 Programming Concepts, K16 Information Processing and

	Presentation, and E1 Computer Networks), and Content (Technology, Secondary 1–3) Learning Outcomes and Remarks (Information and Communication Technology, Secondary 4–6)
Korea	성취기준 [Achievement standards] (Practical Arts (Technology/ Home Economics, Primary 5, 6) 성취기준 [Achievement standards] (Informatics, Lower and Upper Secondary 1–3)
New Zealand	Progress outcome in "Computational thinking for digital technologies" and "Designing and developing digital outcomes" (Technology, Years 1–13)
Poland	Edukacja informatyczna [Information technology education] (Informatics, Grades 1–3) Treści nauczania – wymagania szczegółowe [Teaching content - specific requirements] (Informatics, Grades 4–11)
Portugal	CONHECIMENTOS, CAPACIDADES E ATITUDES [Knolwedge, Capacity, and Attitude] (ICT, Grades 1–9, Computer Applications B, Grade 12)
Sweden	Centralt innehåll [Core content] (Mathematics, Grades 1–9) Centralt innehåll [Core content] (Technology, Grades 1–9) Centralt innehåll [Core content] (Social Studies, Grades 4–9) Core content (Social Studies, Grades 10–12) Central content (Swedish / Swedish as a second language, Grades 10–12) Core content (Computers and ICT, Grades 10–12) Core content (Information and communication, Grades 10–12) Core content (Technology, Grades 10–12)