International Computer Science Standards Comparison Report

Examining the Similarities and Differences in K-12 Computer Science Standards Across Seven Global Locations

CSTA

ACE
INSTITUTE for ADVANCING COMPUTING EDUCATION

Authors & Leadership

The project is primarily planned, facilitated, and coordinated by the *Reimagining CS* project team.

Computer Science Teachers Association

Bryan Twarek (PI)
Jake Koressel (Project Manager)

IACE INSTITUTE for ADVANCING COMPUTING EDUCATION

Dr. Monica McGill (Co-PI)
Dr. Julie Smith (Researcher)

Acknowledgements

We authored this report as an extension of *Reimagining CS Pathways: High School and Beyond,* a community-wide project that explores how CS learning opportunities can be reenvisioned for high school students. CSTA and IACE co-led the project, in partnership with ACM, Code.org, College Board, CSforALL, and ECEP Alliance. This project resulted in a definition of foundational high school computer science content and resulting pathways. View the *Reimagining CS Pathways* report and supporting materials at ReimaginingCS.org.

We extend our gratitude to individuals who assisted with identifying, compiling, and analyzing standards from the locations included in this study: Sofía De Jesús, Chris Stephenson, Gabriel Onifade, Keith Quille, Bruce Fuda, Arto Hellas, Gabriel Onifade, Camilo Vieira Mejía, Alejandro Espinal Duque, and Roisin Faherty.

Funding Support

This project is supported by the National Science Foundation (NSF) under Grant No. 2311746. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

Suggested Citation

CSTA & IACE. (2025). *International Computer Science Standards Comparison Report: Examining the Similarities and Differences in K-12 Computer Science Standards Across Seven Global Locations.* New York, NY: Association for Computing Machinery.

License

Executive Summary

In a time when computing technologies are changing rapidly, computer science (CS) education needs to match that rapid pace in order to prepare students for a world powered by computing. Part of that effort involves developing learning standards that lay the groundwork for curriculum and instruction that meets the needs of students, as well as industry and the broader society. As part of its effort to revise its own standards, CSTA – aided by IACE – studied the K-12 CS standards in six locations around the world in order to compare these standards to CSTA's current standards (the de facto national standards in the US) and therefore gain a better understanding of the international landscape of CS learning standards.

This report presents the results of that analysis. Standards from Nigeria, Ireland, Colombia, Australia, Finland, and Hong Kong were decomposed into smaller elements and then compared. Comparisons explore general trends, such as how standards are organized, what categories of standards are emphasized, and how much coverage of topics varies across locations. Then, we took a deeper dive into CS topic areas (which we organized into ten categories) to gain a better understanding of how that topic is treated across locations. We analyzed how much each topic was emphasized; which elements of standards appeared, how often, and at what grade levels/bands in each location; and distinct aspects of the topic in each location. Throughout the report, we include recommendations for standards writers to consider during the standards revision process.

In general, K-12 CS standards in the locations that we analyzed have a fair amount of overlap, covering topics such as basic programming knowledge, the components of computers, and ethical and social issues related to computing technologies. But there is also a good bit of variation in different locations. Sometimes that variation is in form, such as Ireland's organization of some standards into project-based learning experiences, Australia's well-aligned learning progressions, and Finland's emphasis on playful learning. And sometimes it is in content, such as Nigeria's emphasis on the context and history of computing and Colombia's integration of environmental science topics.

We hope that this report will provide useful insights to those working to develop K-12 CS standards that will help prepare all students to be competent creators, consumers, and citizens in an increasingly technological world.

1. Introduction

Learning standards play an important role: they shape curriculum, instruction, assessment, and, therefore, the student experience. As the Computer Science Teachers Association (CSTA) revises its standards for K-12 computer science, gaining a better understanding of the landscape of standards from around the world can help support that effort by highlighting both similarities and differences between the current CSTA standards and other sets of K-12 CS standards.

This report presents a comparison between the 2017 CSTA K-12 Standards and similar standards used in Nigeria, Ireland, Colombia, Australia, Finland, and Hong Kong. These locations were chosen with an eye toward the selection of a broadly diverse range of locations (in terms of the location's income, demographics, etc.), limited by our ability to access the standards (or a translation of the standards) in English.

Our goal is to explore the landscape of K-12 CS standards across several locations. It is important to note that, when we observe differences between sets of standards, we are not implying that there are "gaps" or deficiencies with any set of standards. It is simply not possible, given the limits of instructional time, to address all possible CS topics – and so including all possible topics in a set of standards would not be advisable.

There are many ways to compare sets of standards; this report uses a few different methods that are most appropriate in light of the report's purpose of informing CSTA's standards revision process. We are primarily concerned with a better understanding of the landscape of CS standards, particularly in terms of topics and articulation across grade bands or levels. We thus used a mixed methods approach in order to better understand the landscape of CS standards internationally, performing both quantitative and qualitative analyses in order to better appreciate the similarities and differences in standards across locations.

The report first provides a brief background on CS education standards in each location. Then, we explain our methodology. Results are shared, organized by CS topic area. Throughout the report, we make recommendations that standards writers may want to consider.

2. Background

2.1 US (CSTA)

We use the 2017 CSTA standards in this project. These standards are used by many U.S. states in the development of their own standards, but the CSTA standards themselves are not officially adopted at the national level.

2.2 Nigeria

Nigeria has articulated standards for students at each grade level (Primary 1 - 6, Junior Secondary 1 - 3, and Senior Secondary 1 - 3) from ages 6 to age 17. Nigeria has featured CS as part of its curriculum for decades, stemming from a 1988 policy on computer education (Tshukudu et al., 2023). The subject is compulsory for all students in the first nine grade levels.

We use the performance objectives for Information Technology and Computers & IT in this project. We added identifiers to the objectives. Nigeria's performance objectives are very granular. In some situations, we assigned multiple related performance objectives to just one identifier and used numbers in parentheses to indicate the original list of performance objectives.

2.3 Ireland

In this project, we use the Technology section from Draft Science, Technology and Engineering Education Specification for Primary, Short Course: Coding for Junior Cycle, and Computer Science Curriculum Specification for Senior Cycle. The Senior Cycle standards include 'ordinary level' and 'higher level' (≈ honours) content. We included both levels of content. To uniquely identify each standard, we added 1, 2, 3, 4, J, or S before each identifier. Some standards (such as 3.3.2) refer to algorithms or programs. These were arbitrarily classified under the Algorithms and CT category (and not the Programming Skills category). We omitted one unrelated standard ("convince their peers that an idea is worthwhile").

2.4 Colombia

We use a translation of the Scope and Sequence Matrix of the Guidelines for the Development of Computational Thinking in this project. Note that this is a set of guidelines used in public schools, developed by the National Ministry of Technologies, the National Ministry of Education, and the British Council. We added identifiers to the standards.

2.5 Australia

CS instruction (termed Digital Technologies) before the tertiary level was launched at scale in Australia in 2014 (Bell et al., 2024) and is now compulsory across eleven years of instruction (Webb, 2017).

Australia's current standards contain, in addition to the standards themselves, an "achievement standard" for each grade band, which summarizes what the student should know and be able to do at the end of the band. Australia also provides samples of student work related to these standards, categorized as above satisfactory, satisfactory, and below satisfactory.

We use Australia's Digital Technologies standards in this project.[1]

2.6 Finland

At the K-12 level, Finland does not have CS as a separate subject (Malmi et al., 2023); it is integrated in other areas (Mäkitalo et al., 2024), including language courses (Niemelä et al., 2022).

We use the Programming Competence section of the Framework for Digital Competence in this project. We omitted a few items at the lower levels that were not technology specific, and we added identifiers to the standards.

2.7 Hong Kong

At the primary level, Hong Kong has focused on computational thinking education – and focused its attention on grades 4 - 6 (also known as Stage 2) for this effort (Kong & Kwok, 2024). The instruction is not compulsory but is recommended. We note that Hong Kong's secondary standards are unusual in that they usually do not include verbs (e.g., "distinction between random access memory (RAM) and read only memory (ROM)," without indicating whether students should *investigate*, *understand*, *describe*, or *explain* this distinction).

We use Computational Thinking – Coding Education for Primary and Information and Communication Technology standards from Technology Education for Secondary in this project. We added identifiers to the standards, and we omitted a few unrelated standards (e.g., business communication). Some topics in these standards were labelled as "extension" – we did include these in the dataset.

3. Methods

This project analyzes the CS standards from seven different locations: the US, Nigeria, Ireland, Colombia, Australia, Finland, and Hong Kong. We used the set of standards most closely associated with CS for each location, and we note that differences in how locations organize their standards results in some inconsistencies between locations. For example, Finland does not articulate CS standards for general upper secondary education, but it does include some algorithmic thinking in its math standards. We do not include these math standards since they are articulated as part of math and not CS (despite the fact that the content is very similar to what is articulated in CS standards in other locations); we also do not include the standards for Finland's vocational qualification as a Software Developer (since it is not part of 'general' education).

We also note that there is a difference – sometimes a profound difference – between the content found in a set of learning standards and the 'enacted curriculum,' or what actually happens in the classroom. For example, Nigeria's standards refer to the programming language BASIC, but classroom instruction in Nigeria sometimes uses other programming languages, such as Scratch or Python. This project focuses strictly on the written learning standards; we did not explore the question of how students' classroom experiences differ from what is described in the learning standards.

To begin our analysis, we divided the standards into elements. An element is a discrete portion of a standard. For example, consider CSTA 3A-AP-22: "Design and develop computational artifacts working in team roles using collaborative tools." There are two distinct elements in this standard: *designing* and *developing*. We began by dividing the CSTA standards into elements. Then, as the standards from other locations were analyzed, we added to or modified the elements as needed, iteratively developing a list of elements to represent the full range of content of the standards in a manner more atomic and granular than the standards themselves, which often combine multiple distinct elements. Throughout this process, we had to balance the benefits of a larger set of elements (e.g., more specificity and precision) with the drawbacks (e.g., making comparative analysis less robust). The end result was a list of 93 elements.

Categories
Algorithms and CT
Data
Digital Literacy
Human Computer Interaction
Impacts of Computing
Networking
Programming Skills
Security
Software Development
Systems

These elements were arranged into ten categories for analysis (see table left). We note some overlap, gaps, and ambiguities in these elements, categories, and the assignment of standards to them. For example, three Australian standards concern a student's digital footprint (AC9TDI6P10, AC9TDI6P10, and AC9TDI10P14); these might have been classified under Data, Digital Literacy, Impacts of

Computing, or even Networking. (We somewhat arbitrarily chose Impacts of Computing.) Similarly, the distinction between Programming Skills and Software Development is perhaps particularly thin; however, we conceptualize Programming Skills as more technical and lower level, with broader and higher-level content assigned to the Software Development category.

For each standard from each of the seven locations included in this project, we recorded the following information: the location of origin, the grade level/band, the identifier, the text of the standard, the category, and the element(s) found in that standard. In addition to these data points, a qualitative analysis was conducted, focusing on the content of each standard as well as broader observations about what was (and was not) emphasized in each location.

While decomposing standards into elements was a necessary precursor to analyzing the standards, we note that there are often important differences between standards that contain the same element. For example, the element “model how hardware and software work together” covers these two standards: “describe [a] computer as Input — Process — Output (IPO) system” (Nigeria, N4.4) and “creates and programs a prototype to control elements at a distance” (Colombia, C8.10). These two standards are quite different in terms of what they expect students to know and do, both in terms of their content and in terms of their cognitive complexity. We also note that we only included elements that were *explicitly* referenced in a standard. It is likely the case that many other elements were intended to be *implicitly* included in the standards, but we chose not to include these due to the subjective nature of the determination that would be required.

The table on page 10 shows the grade levels/bands included in the dataset. It also shows how we mapped the levels/bands of other locations to the US bands. We performed this mapping to enable comparison across levels/bands. The bands do not align perfectly across locations (e.g., Ireland’s Junior Cycle straddles US 6-8 and 9-10), but we attempted to align them as closely as possible.

<table>
<tr><th>Age</th><th>US</th><th>Nigeria</th><th>Ireland</th><th>Colombia</th><th>Australia</th><th>Finland</th><th>Hong Kong</th></tr>
<tr><td>4</td><td></td><td></td><td rowspan="2">Stage 1 (2)</td><td></td><td></td><td rowspan="2">ECEC (10)</td><td></td></tr>
<tr><td>5</td><td rowspan="3">K-2 (18)</td><td></td><td>Grade 0 (6)</td><td>Foundation (3)</td><td></td></tr>
<tr><td>6</td><td>Primary 1 (3)</td><td rowspan="2">Stage 2 (2)</td><td>Grade 1 (15)</td><td rowspan="2">Years 1-2 (9)</td><td>Pre-Primary (10)</td><td></td></tr>
<tr><td>7</td><td>Primary 2 (2)</td><td>Grade 2 (9)</td><td rowspan="2">Grades 1-2 (10)</td><td></td></tr>
<tr><td>8</td><td rowspan="3">3-5 (21)</td><td>Primary 3 (4)</td><td rowspan="2">Stage 3 (2)</td><td>Grade 3 (13)</td><td rowspan="2">Years 3-4 (12)</td><td></td></tr>
<tr><td>9</td><td>Primary 4 (5)</td><td>Grade 4 (12)</td><td rowspan="3">Grades 3-6 (13)</td><td rowspan="3">Stage 2 (26)</td></tr>
<tr><td>10</td><td>Primary 5 (3)</td><td rowspan="2">Stage 4 (2)</td><td>Grade 5 (14)</td><td rowspan="2">Years 5-6 (14)</td></tr>
<tr><td>11</td><td rowspan="3">6-8 (23)</td><td>Primary 6 (6)</td><td>Grade 6 (18)</td></tr>
<tr><td>12</td><td>Junior Secondary 1 (5)</td><td rowspan="3">Junior Cycle (24)</td><td>Grade 7 (15)</td><td rowspan="2">Years 7-8 (18)</td><td rowspan="3">Grades 7-9 (13)</td><td>Secondary 1 (26)</td></tr>
<tr><td>13</td><td>Junior Secondary 2 (13)</td><td>Grade 8 (14)</td><td>Secondary 2 (16)</td></tr>
<tr><td>14</td><td rowspan="2">9-10 (30)</td><td>Junior Secondary 3 (8)</td><td>Grade 9 (14)</td><td rowspan="2">Years 9-10 (17)</td><td>Secondary 3 (15)</td></tr>
<tr><td>15</td><td>Senior Secondary 1 (15)</td><td rowspan="3">Senior Cycle (59)</td><td>Grade 10 (16)</td><td></td><td></td></tr>
<tr><td>16</td><td rowspan="2">11-12 (28)</td><td>Senior Secondary 2 (14)</td><td>Grade 11 (14)</td><td></td><td></td><td></td></tr>
<tr><td>17</td><td>Senior Secondary 3 (10)</td><td></td><td></td><td></td><td></td></tr>
<tr><td colspan="8">Note that the numbers in parentheses indicate how many standards there are at each grade band/level.</td></tr>
</table>

Note that some locations do not articulate CS standards at some bands/levels (e.g., Colombia grade 12). This fact should be kept in mind in the analysis that follows, especially where comparisons are made.

We wanted to explore the cognitive complexity of the standards. To do this, verb counts for each location were determined using the Python library spacy. We attempted to exclude verbs unlikely to represent what a student is expected to do (e.g., an infinitive indicating the purpose of an activity such as "address" in CSTA 2-AP-10: "use flowcharts and/or pseudocode to address complex problems as algorithms"). However, the process is not perfect, and so the results are only an approximation. Note also that many of Hong Kong's standards do not include verbs.

4. Findings

4.1 General Trends in Content

4.1.1 Standards Organization and Other Broad Issues

Granularity differs widely by location. Nigeria's standards are described at the lowest level of granularity of all of the locations, and they provide detailed information about what students should know and do. For example, Nigeria N1.1 reads: "(1) describe a computer. (2) name parts of a computer. (3) identify computer parts. (4) state the differences between a computer and a television. (5) draw/sketch parts of a computer." At the other end of the spectrum, Finland's standards are articulated at a very high level of abstraction, such as Finland F34: "is able to work on a digital product containing a narrative or elements of gaming using animation or simple programming under guidance or in collaboration with others." Granularity can also differ widely within one location's standards, such as Hong Kong; compare "generate and print a set of integers using [a] random number generator to observe randomness of the pattern" (Hong Kong, H73) with "error detection by verification and validation" (Hong Kong, H50). There are advantages and disadvantages to having greater/lower levels of granularity: more granular standards may provide more consistency in implementation and more scaffolding for teachers, but more granular standards may also be less flexible.

Organization patterns differ widely. Organizing standards into logical groups and then determining alignment across grade levels/bands and across groups is a complicated task. Consider the charts showing how each location organizes its standards (see Appendix A). Some locations use entirely different organization systems at different levels/bands (e.g., Ireland), which may promote content designed for the distinct needs of each band. Other locations use the same organizational system across levels/bands (e.g., the US), which may promote curricular cohesiveness. Similarly, some locations have instances where a category is not addressed at some levels (e.g., Colombia), which may reflect careful tailoring of standards to the needs of an age group, as well as parsimony. Other locations cover all categories at all levels (e.g., Finland), which may permit better alignment and progression of concepts as students age.

The degree of alignment of learning objectives across grade levels/bands differs by location. Consider these standards from Australia, which has an unusually high degree of alignment across grade bands for some standards, as the table on page 12 shows (modifications from the previous level are underlined).

Grade Band	Text
Years 3-4	generate, communicate and compare designs (Australia, AC9TDI4P03)
Years 5-6	generate, modify, communicate and evaluate designs (Australia, AC9TDI6P04)
Years 7-8	generate, modify, communicate and evaluate alternative designs (Australia, AC9TDI8P08)
Years 9-10	generate, modify, communicate and critically evaluate alternative designs (Australia, AC9TDI10P08)

In other locations, standards appear to be largely independent across grade level/bands, with some topics addressed only once. For example, Colombia has a standard with the element "explore how information travels over networks" (Colombia, C8.8) at only one grade level, year 8.

AI – although not widely covered currently – is likely to be in the future, and organizing/locating it in standards presents some challenges. Note that, in our dataset, AI is referenced explicitly in two elements: "explore AI systems" in the Systems category and "explore AI algorithms" in the Algorithms and CT category. References to AI also appears in standards with other elements:

- "develop a program"
- "evaluate programs re: their effects on society"
- "appreciate the role of data"
- "explore IT and IT devices"

AI may also be involved in the implementation of other elements even when it is not specifically mentioned, including "transform/organize data" and "explore data privacy issues," among others. We note that Australia has articulated a "curriculum connection" for AI for all years, with guidance for integrating AI-related content into other curriculum areas (these connections are not included in this project). Deciding whether AI-related learning content should form its own category within CS standards, be integrated into other categories of CS standards, and/or be integrated across all subjects is a difficult decision, especially given the rapid evolution of AI technologies.

Locations provide different types of supplementary information. Locations include various supplementary information alongside their standards. For example, Ireland provides:

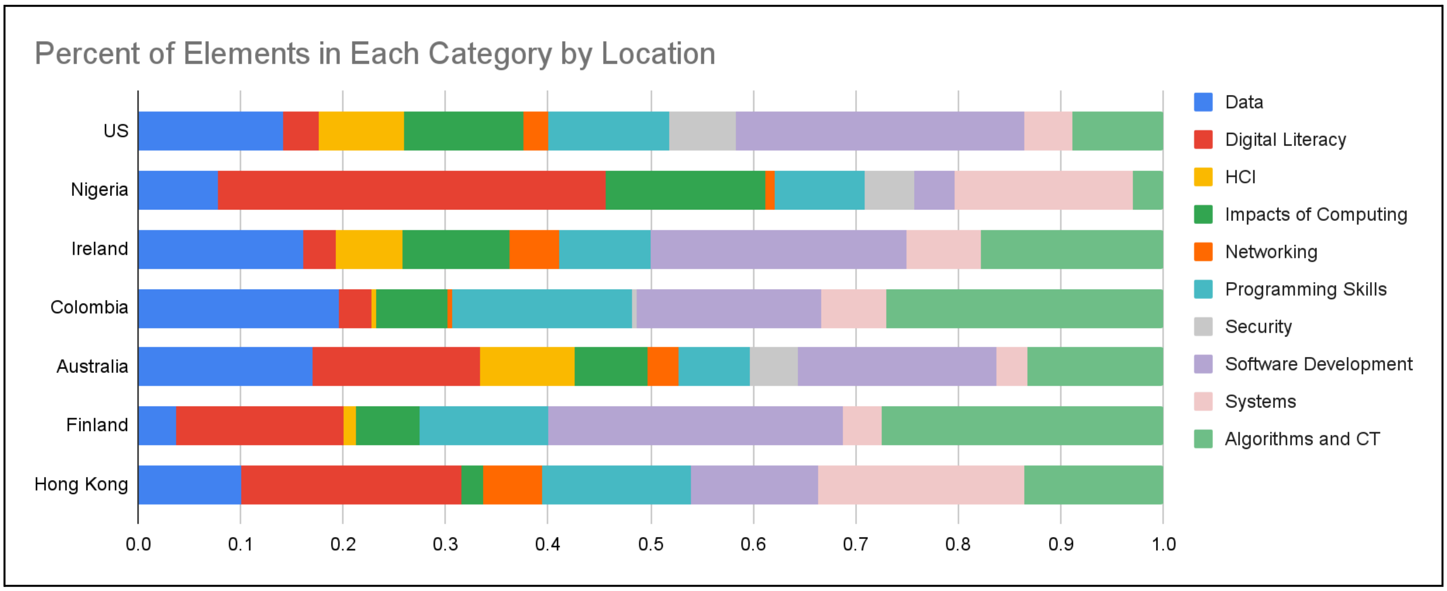
- guidance on assessment, including assessment criteria
- time requirements (i.e., Senior Cycle standards are designed for 180 hours of class time)
- brief details about standards; e.g., for "apply basic search and sort algorithms . . .", Ireland, S2.8, there is a list for sorting (simple sort, insert sort, bubble sort, quick sort) and searching (linear search, binary search)

4.1.2 Emphasized Categories

The figure Percent of Elements in Each Category by Location (see below) shows how common each category is in each of the seven locations. Overall, there is wide variation across locations. For example, in Nigeria, <4% of elements fit into the Software Development category (represented by the light purple bars), but in Ireland, 25% of elements are in this category. Similarly, about 12% of US elements are in Impacts of Computing (dark green bars), but <3% of elements from Hong Kong are. Some categories are not covered at all in some locations:

Recommendation: Analyze drafts of proposed standards to ensure that coverage of categories aligns with identified priorities.

- HCI in Nigeria
- Security in Ireland
- Networking and Security in Finland
- HCI and Security in Hong Kong



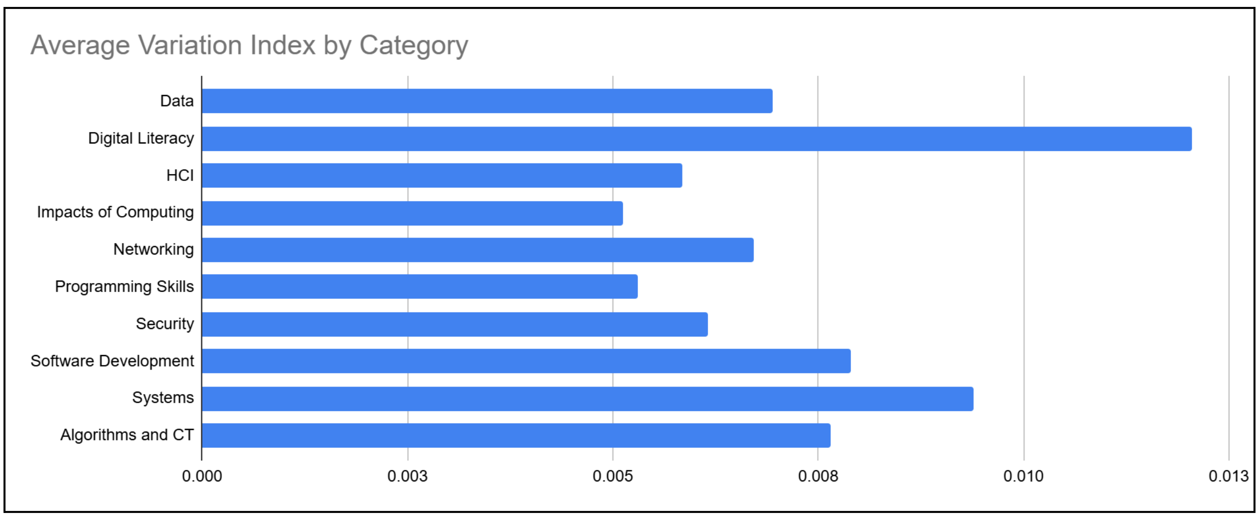
We note one trend: Nigeria and Hong Kong emphasize the Digital Literacy and the Systems categories, each with >40% of their elements from two categories. In contrast, no other location had >20% of their elements from these two categories (see the table below).

Percent of Digital Literacy + Systems Elements	Location
55%	Nigeria
42%	Hong Kong
20%	Finland
19%	Australia
10%	Ireland
10%	Colombia
8%	US

We also note that some (but not all) of Ireland's Senior Cycle (≈ 11 - 12 grade) learning standards are organized into four applied learning tasks, each of which constitutes a cohesive project: (1) developing a website with a database, (2) analyzing data, (3) developing a model, and (4) working with a physical computing system.

Recommendation: Consider grouping standards to promote project-based learning.

4.1.3 Variation in Elements by Category



As the figure Average Variation Index by Category (see above) shows, Digital Literacy is by far the category with the highest average variation index (0.012). In other words, elements within the Digital Literacy category had the widest variation in how often they occur in different locations. For example, "operate software" was the most commonly occurring of all of the elements, with 59 instances, which was >6% of all elements. In the US, "operate software" was <1% of elements, while in Nigeria, it was >12%. Thus, locations vary greatly in the extent to which using different kinds of software is covered in their CS standards. By contrast, the two categories with the smallest variation indices were Impacts of Computing (~0.005) and Programming Skills (~0.005). These lower values suggest that elements in these categories are covered more evenly across locations than the Digital Literacy elements are. The table below shows the elements with the highest variation indices.

Element	**Category**	**Variation Index**
Operate software.	Digital Literacy	0.047
Identify and describe the function of components of computers.	Systems	0.031
Explore IT and IT devices.	Digital Literacy	0.026
Develop a program.	Software Development	0.024
Identify abstractions.	Algorithms and CT	0.021

The table below shows the elements that occur at least 20 times.

Element	**Category**	**Count**
Operate software.	Digital Literacy	59
Develop a program.	Software Development	42
Identify and describe the function of components of computers.	Systems	38
Debug an algorithm, program, or device.	Software Development	27
Create algorithms.	Algorithms and CT	26
Identify abstractions.	Algorithms and CT	26
Explore IT and IT devices.	Digital Literacy	25
Design a program.	Software Development	24
Explore data representation.	Data	23

4.1.4 Cognitive Complexity

Recommendation: Ensure that the cognitive complexity of standards aligns with what writers have determined is most appropriate.

The next tables (see below) show the most common verbs by location. It suggests what kinds of verbs – and, therefore, what level of cognitive complexity – is most common in each location. Note, however, that verb choice is not a perfect proxy for cognitive complexity (e.g., *identifying* a bug and *identifying* an input device are not of the same complexity). Nonetheless, the chart below – as well as the qualitative analysis – support the idea that there is some variation in the cognitive complexity across locations. Nigeria's standards are largely focused on lower-order thinking skills, perhaps reflecting their focus on Digital Literacy content. In contrast, other locations emphasize more higher-order thinking skills.

Note: in the two tables below, the shading of the verbs indicate its Bloom's level, with darker shading indicating higher levels of Bloom's.

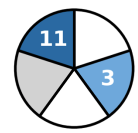
US			Nigeria			Ireland			Colombia		
Verb	**Count**	**%**	**Verb**	**Count**	**%**	**Verb**	**Count**	**%**	**Verb**	**Count**	**%**
use	11	5.7%	identify	33	13.3%	explain	12	7.3%	uses	25	11.7%
evaluate	10	5.2%	define	31	12.5%	use	11	6.7%	recognizes	19	8.9%
develop	10	5.2%	list	23	9.3%	describe	11	6.7%	explains	13	6.1%
explain	8	4.1%	describe	21	8.5%	identify	9	5.5%	identifies	8	3.8%
describe	7	3.6%	mention	13	5.2%	explore	5	3.0%	creates	7	3.3%

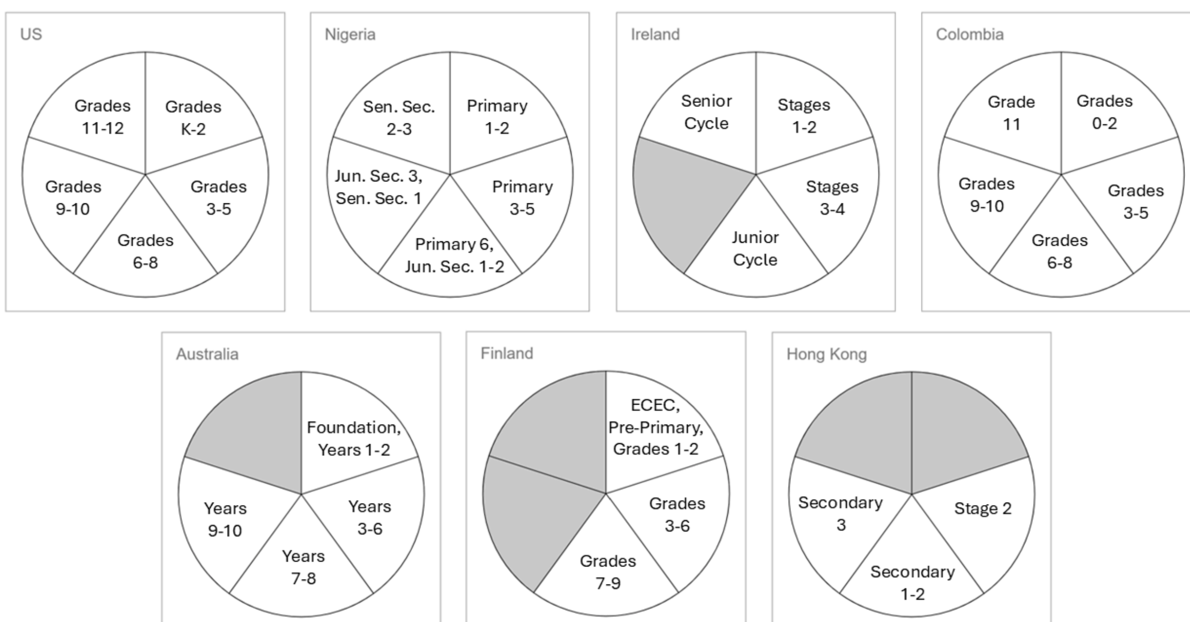
Australia			Finland			Hong Kong		
Verb	**Count**	**%**	**Verb**	**Count**	**%**	**Verb**	**Count**	**%**
use	10	6.6%	describe	4	2.8%	include	8	11.0%
represent	8	5.3%	work	4	2.8%	understand	6	8.2%
explain	8	5.3%	familiarise	4	2.8%	generate	3	4.1%
explore	7	4.6%	practise	3	2.1%	recognise	2	2.7%
investigate	7	4.6%	name	3	2.1%	modify	2	2.7%

4.2 Trends by Topic Area

In this section, there are three different data visualizations for each topic area:

1. **Percent of elements (with a green border):** this chart shows, for each location, what percent of its elements fit within this topic area.
2. **Coverage of elements by US grade band (with a grey border):** this figure contains a grid of pie charts, one chart for each element for each location. Within each pie chart, grey wedges indicate that the location does not articulate CS standards at this grade band. White wedges indicate that the location *does* articulate CS standards at this grade band, but does not include this element at this band. The wedges in shades of blue indicate that the location does include this element at this grade band. The numbers on blue wedges indicate the starting grade level of the band. So, for example, the pie chart to the right shows that this location has a given element at the 3-5 and the 11-12 grade bands. It does not have the element at the K-2 or the 6-8 grade band (although it does have other CS standards for these bands), and it does not have any CS standards at the 9-10 grade band. The figure below shows which portion of each pie chart map to specific grade levels/bands in each location; grey wedges indicate that no standards are articulated at that grade band.



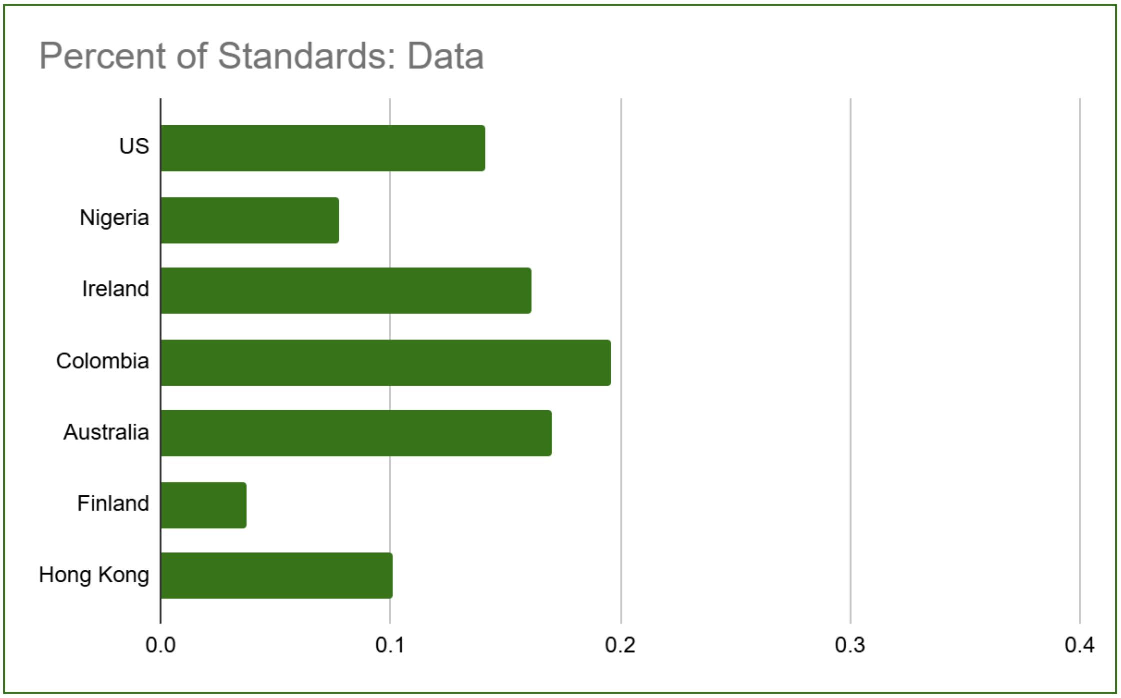


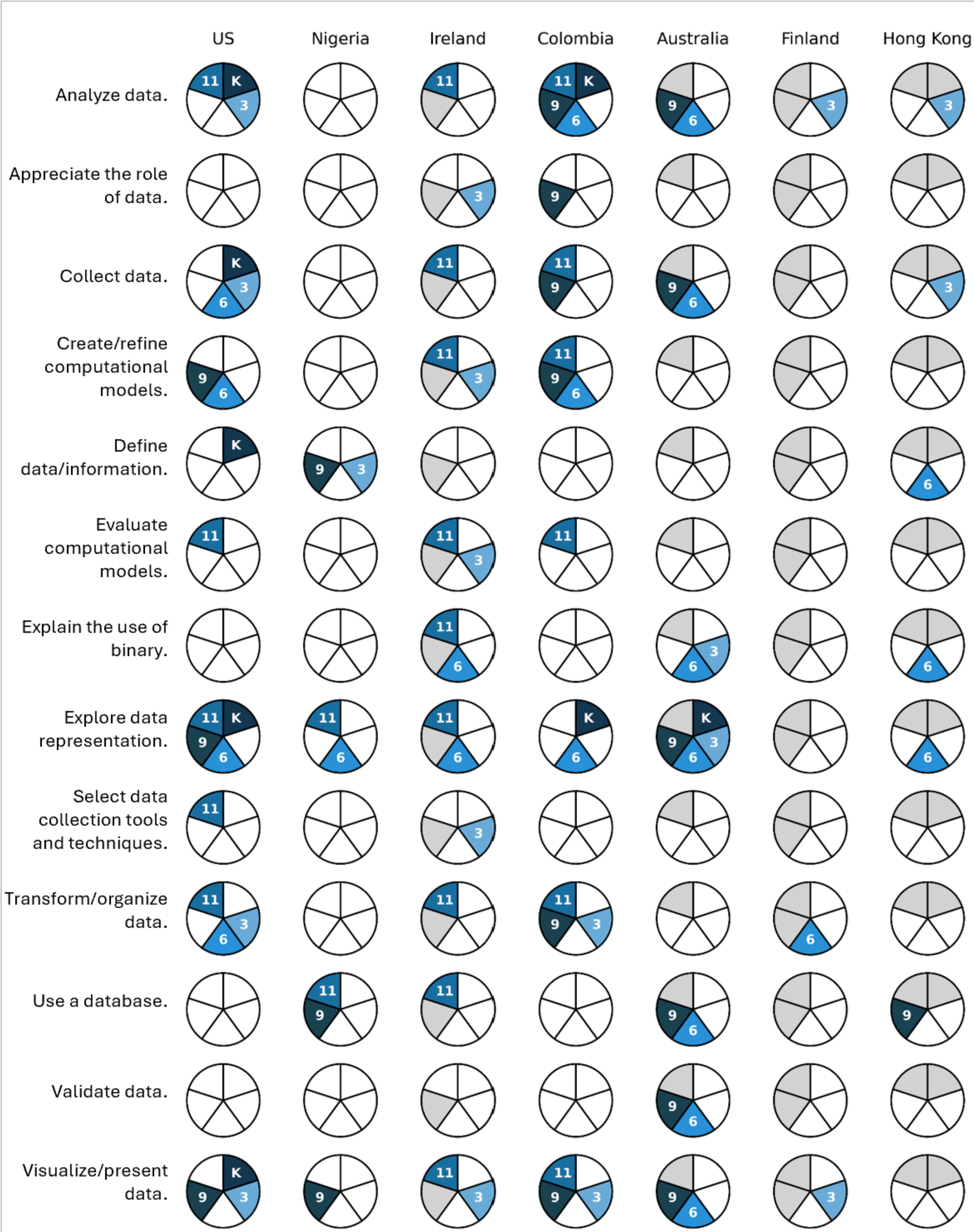
3. **Variation index (with a blue border):** this chart shows, for each element, its variation index. We wanted to be able to compare how much coverage of an element differed across locations. To do this, we use a metric that we call the variation index. A low variation index suggests that locations are relatively similar in how common this element

is in their standards. See Appendix C for an explanation of how we calculated this metric. The variation index ranged from 0.001 to 0.047.

4.2.1 Data

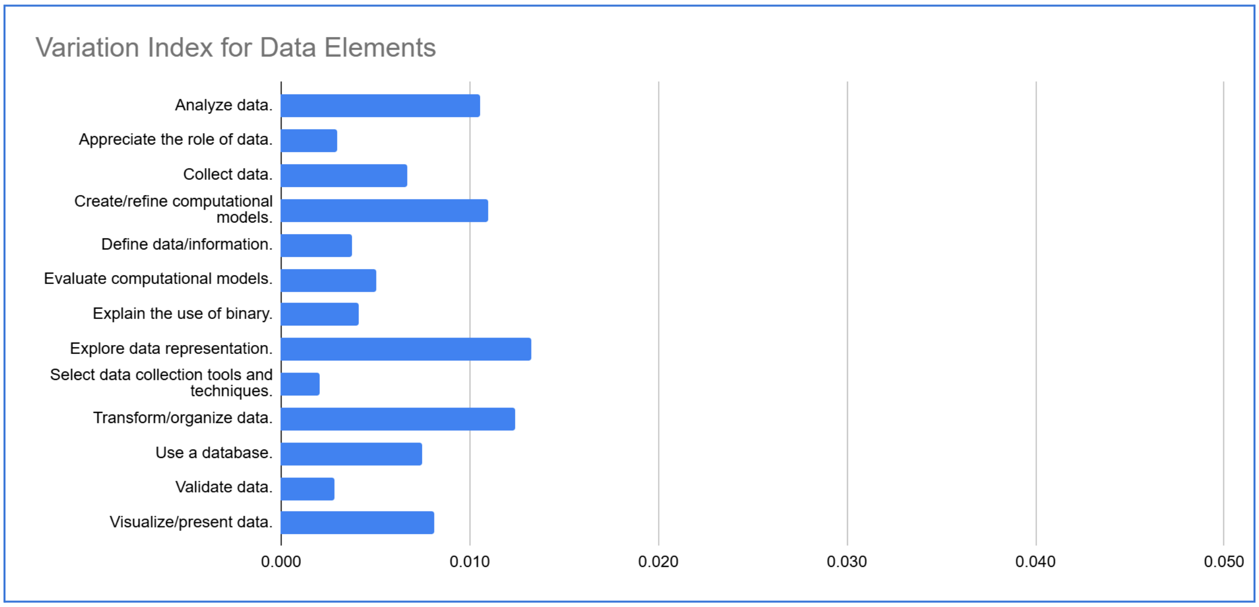
As the figure Percent of Standards: Data shows, locations range from <4% (Finland) to ~20% (Colombia) in the proportion of their elements that fit into the Data category.




US
Nigeria
Ireland
Colombia
Australia
Finland
Hong Kong
Analyze data.
Appreciate the role of data.
Collect data.
Create/refine computational models.
Define data/information.
Evaluate computational models.
Explain the use of binary.
Explore data representation.
Select data collection tools and techniques.
Transform/organize data.
Use a database.
Validate data.
Visualize/present data.

As the chart on page 19 shows, there is a wide range in terms of which elements are covered, and at which grade bands. The most common element in the Data category is “explore data representation,” with 23 instances. It is likely that future iterations of standards across locations will include more data-related content, given the increasing prominence of data science in K-12 (Forstag, 2023), including elements of AI.

The chart below, Variation Index for Data Elements, shows that the most variance exists for “explore data representation.” This element contains a relatively wide range of learning topics, including everything from exploring the processing of Chinese characters (Hong Kong, H37) to image representation (Ireland, J2.6) to data compression techniques (Australia, AC9TDI10K03). Elements with the least variation, such as “appreciate the role of data” and “select data collection tools and techniques,” tend to occur rarely.



Recommendation: Use CS tools and skills to meet learning objectives for other subjects.

A distinctive feature of Colombia’s data-related standards is that three of them incorporate environmental science learning objectives: “simulates how an ecosystem can be sustainable or not” (Colombia, C9.10), “identifies ways in which computation can be helpful in solving complex problems like the state of the environment” (Colombia, C9.13), and “explains what the sustainable development goals are and recognizes how the data allow for tracking their achievement” (Colombia, C10.16). These standards represent an interesting mapping of learning objectives from other subject areas into CS education.

Recommendation: Center equity via examples, especially in supplementary materials.

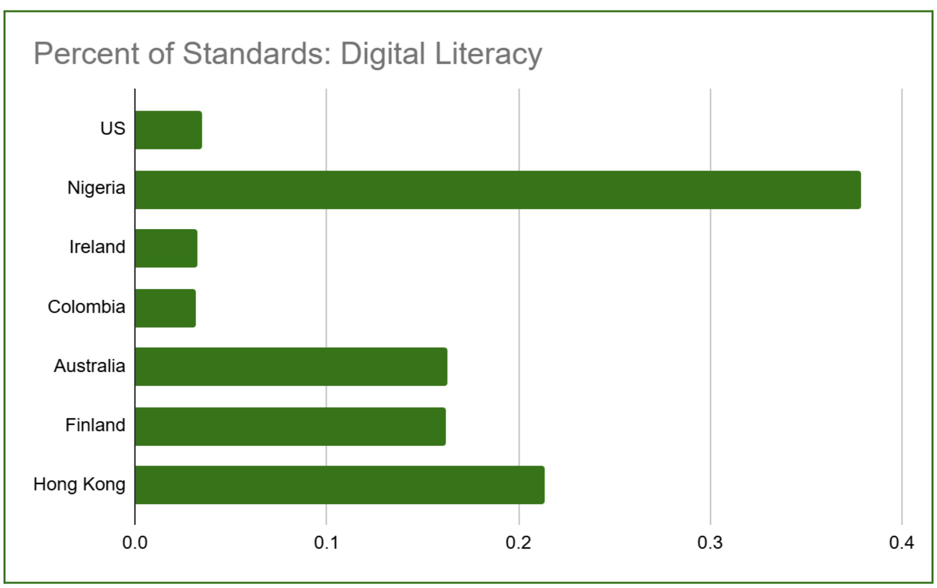
A notable feature of Australia's standards is that each one is accompanied by what are called content elaborations, which contain a few bullet points prefaced by the phrase "this may involve students." These elaborations are helpful in clarifying the content. They also are frequently used to foreground the experiences of and knowledge about First Nations Australian communities. For example, the Year 3-4 standard "define problems with given design criteria and by co-creating user stories" (Australia, AC9TDI4P01) contains this elaboration: "developing a problem statement for collecting and managing information, for example how First Nations Australian rangers could monitor animal populations, such as local marine turtles, using global positioning systems (GPS)."

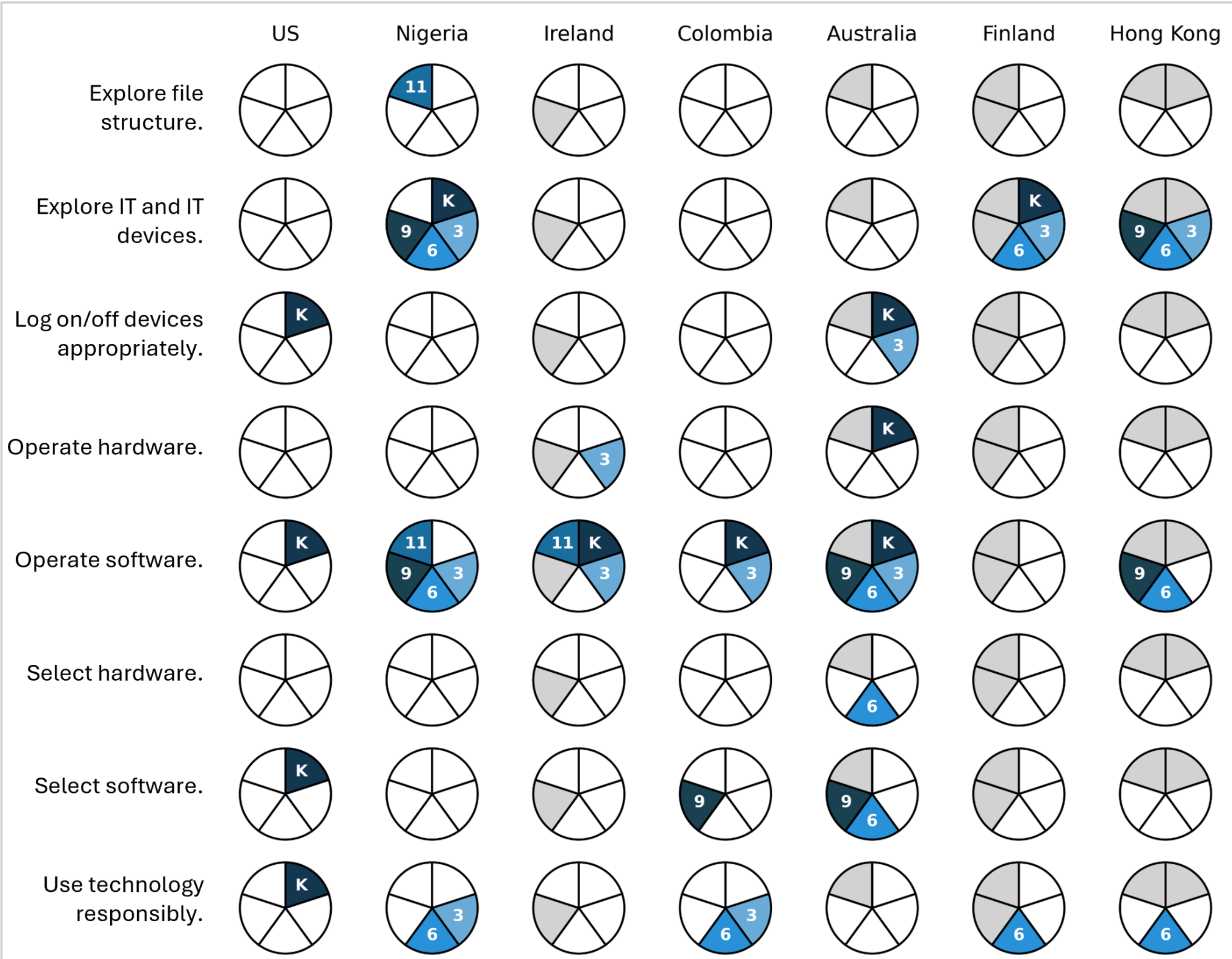
4.2.2 Digital Literacy

As the figure Percent of Standards: Digital Literacy shows, over one-third of Nigeria's standards are focused on digital literacy, with many instances of the elements "explore IT and IT devices" and "operate software" across most grade bands. In contrast, many other locations either do not include these elements (e.g., "operate software" in Finland) or only include them at the lower grade bands (e.g., "operate software" in the US).

Recommendation: Consider whether and how Digital Literacy should be included in CS standards, other subjects, and/or integrated across subjects.

While >16% of Australia's standards fit into the Digital Literacy category, Australia also articulates a separate "general capability" for digital literacy, which is designed to be integrated into other school subjects. (This organization exists because each general capability has a 'home' location, and Digital Literacy's is Digital Technology. Content from the general capability was not included in this project.)



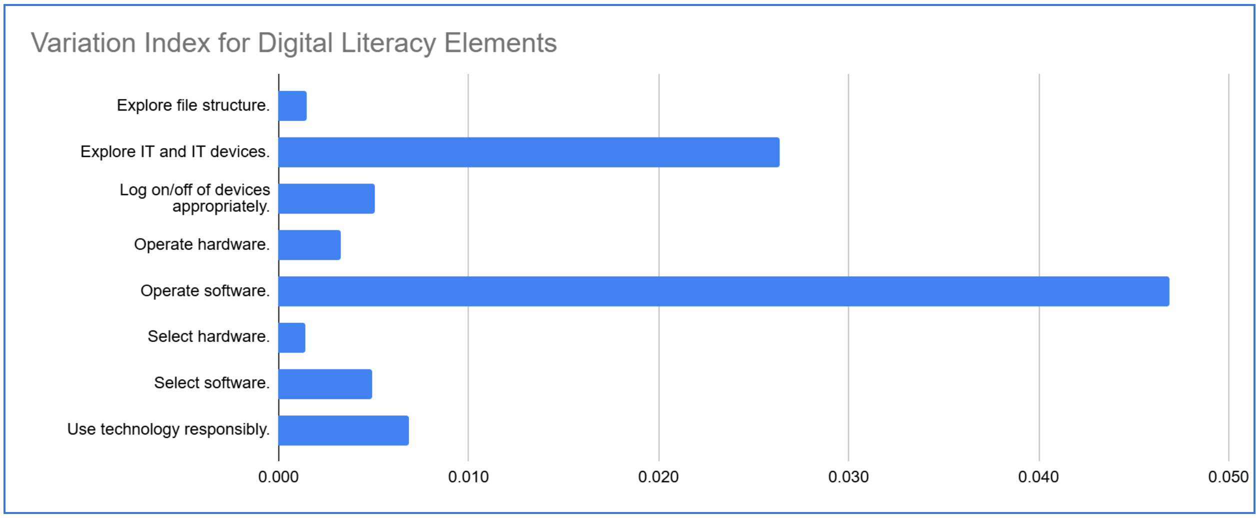


As the pie charts above suggest, there is quite a bit of variation in which elements are covered at which levels in the Digital Literacy category. For example, despite being the most common element overall, “operate software” is not included at all in Finland’s standards, and is only in the K-2 grade band in the US.

The pie chart also shows that, in the US, Digital Literacy elements are restricted to the K-2 grade band. In contrast, in other locations, these elements span most or all grade bands.

Recommendation: Determine whether skills not included previously, such as understanding how file structure works, should be included in response to changes in user behavior.

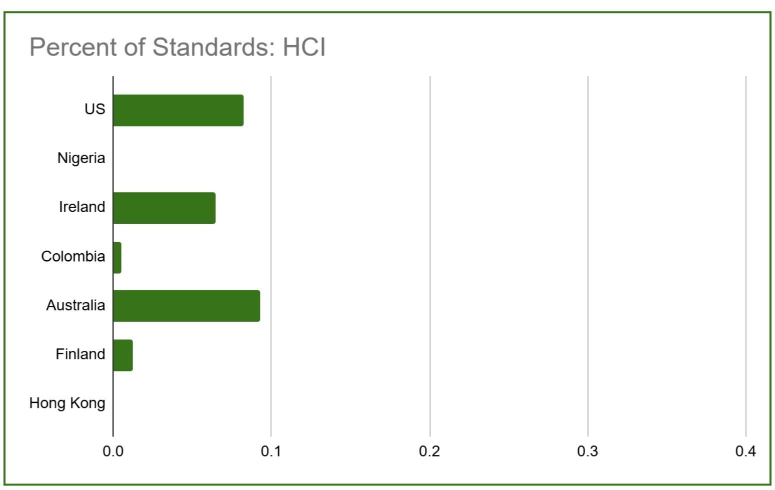
The element “explore file structure” occurs just once (Nigeria, NS2.6). Recently, there has been some concern that students do not know how file structure works (Chin, 2021), perhaps due to improved search features and/or primarily using Chromebooks.

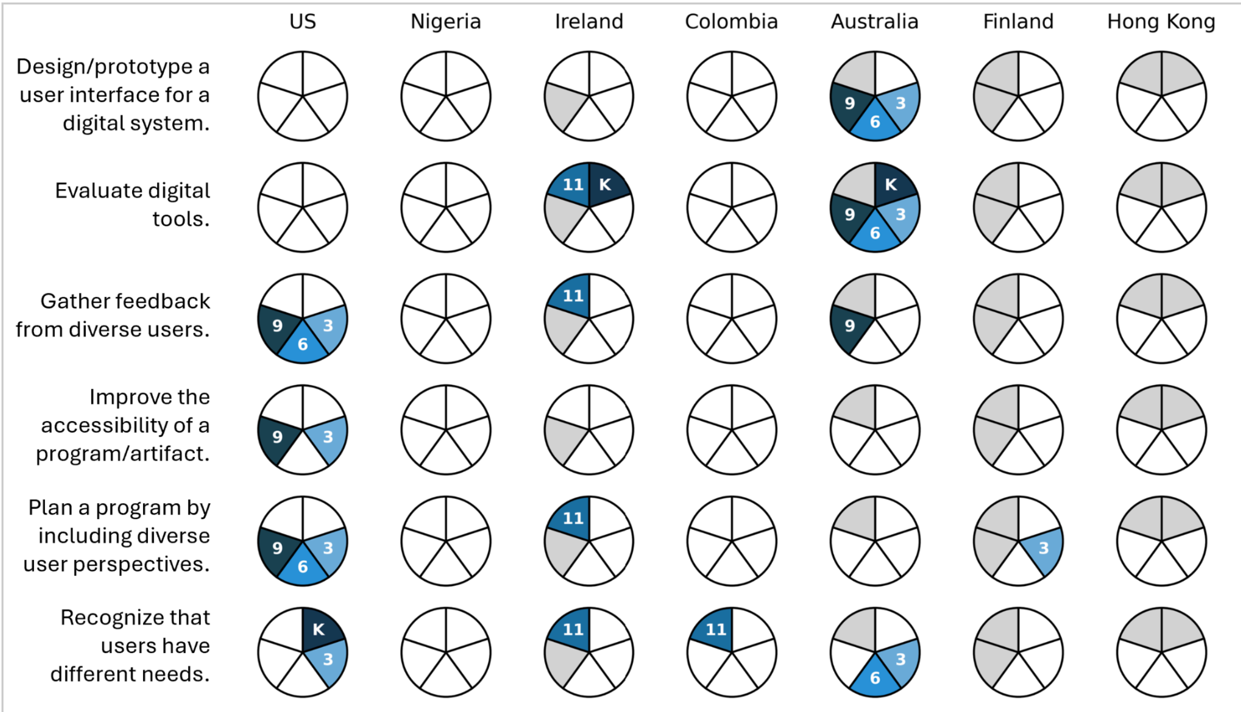


The largest variation index for all elements is for "operate software" (0.047) (Note that standards related to using database software appear in the Data category, and a standard related to using IDEs appears in Software Development.) As described above, this element is also the most common of all of the elements. But it does not appear at all in Finland's standards, and it occurs only three times for Colombia. Many of the standards with this element refer generically to operating software (e.g., "Select and operate appropriate software to perform a variety of tasks . . .", US, 1A-CS-01) or describe a broad range of software (e.g., "select and use appropriate digital tools effectively to share content online, plan tasks and collaborate on projects, demonstrating agreed behaviours," Australia, AC9TDI6P08). Other instances specify the type of software, including office software, search engines, word processors, spreadsheets, image/video editors, learning management systems, email, and so forth.

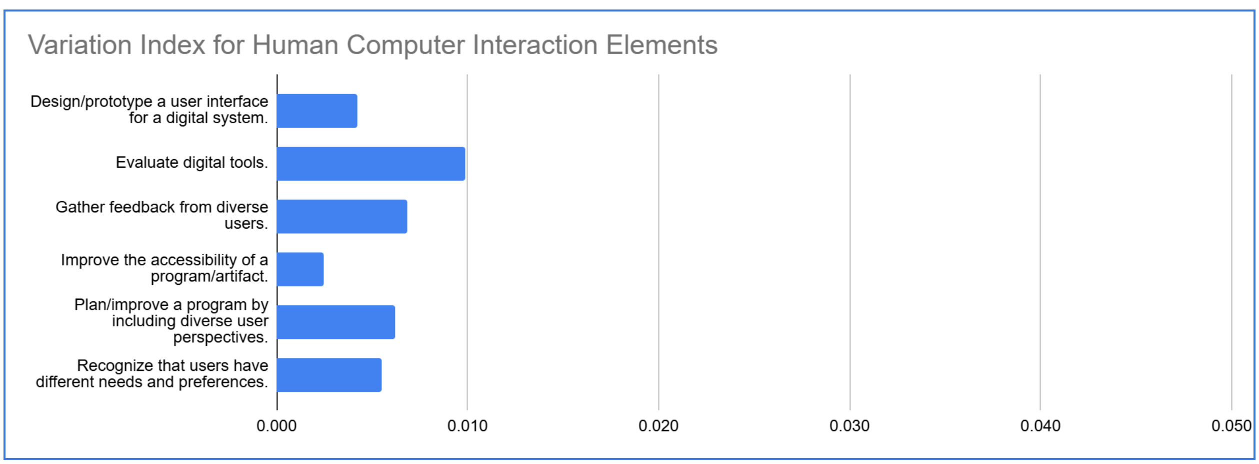
4.2.3 Human Computer Interaction

As the figure below, Percent of Standards: HCI, indicates, Nigeria and Hong Kong do not have any HCI-related elements (Colombia and Finland each have only one). In no location are HCI elements >10% of the total of elements.





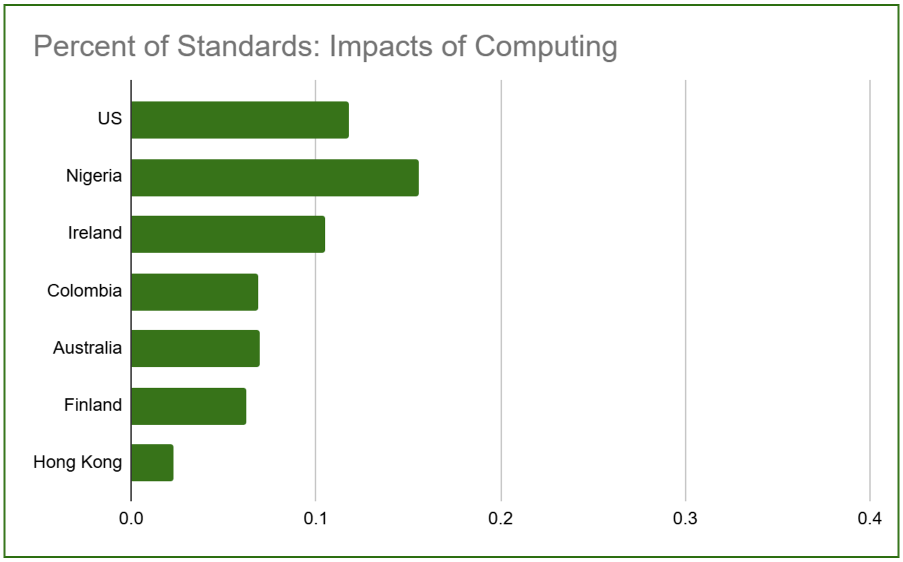
As the pie charts above illustrate, HCI elements tend to occur in the highest grade band in Ireland and in the middle grade bands in the US and Australia.



Recommendation: Consider including standards involving evaluating digital tools, a task adults are likely to do regularly; it also provides opportunities to explore key elements identified by the *Reimagining CS Pathways* project (e.g., reflectiveness, critical thinking, impacts and ethics, inclusive collaboration, and human-centered design).

The figure above, Variation Index for HCI Elements, shows that "evaluate digital tools" has the most variance: it appears in every Australian grade band, but in no other location save Ireland. The ability to thoughtfully and critically evaluate digital tools is likely to be required widely by adults, who – regardless of career path or personal interests – are likely to need the ability to determine whether and how to use a specific digital tool.

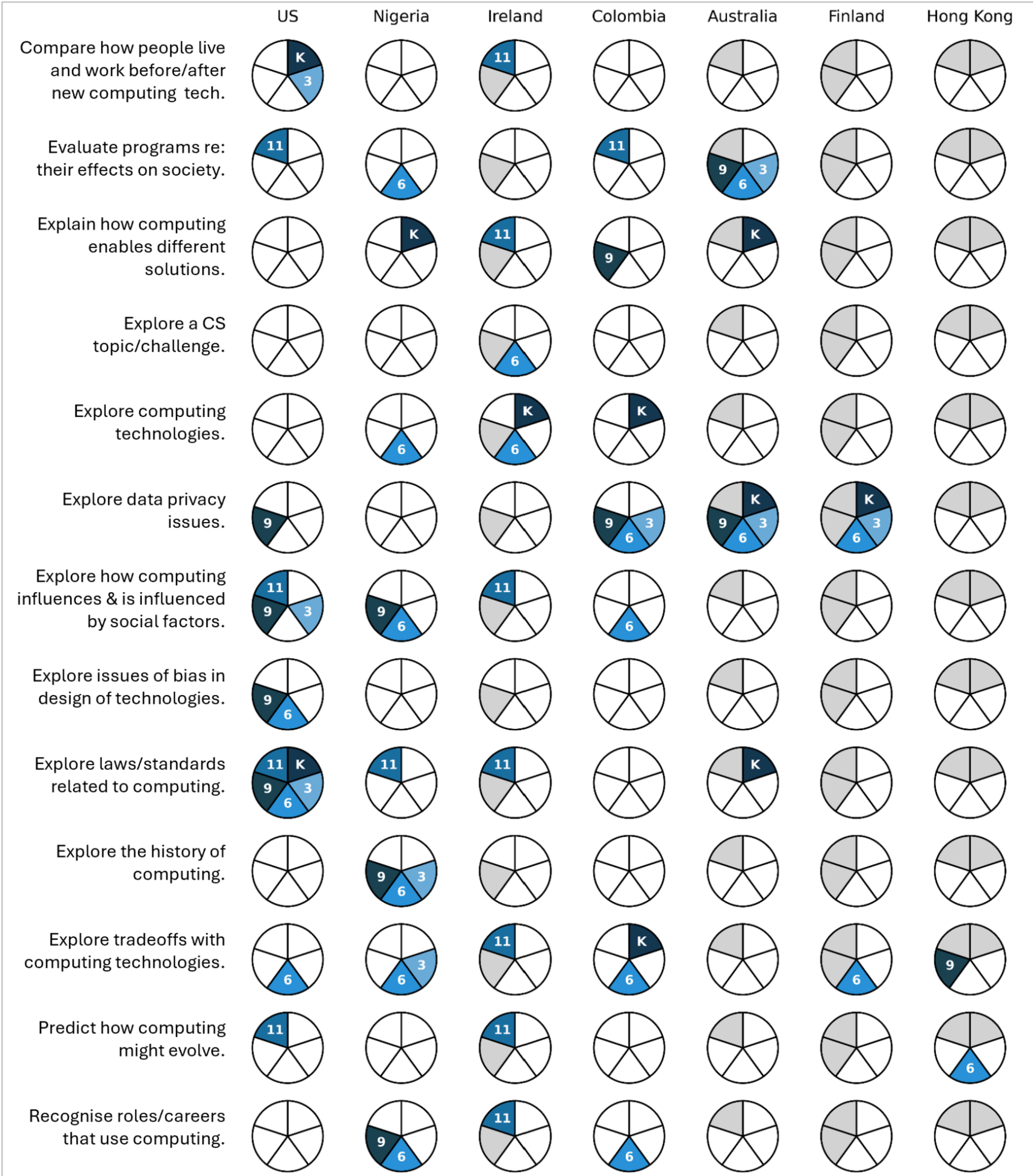
4.2.4 Impacts of Computing



As the figure above, Percent of Standards: Impacts of Computing, shows, Nigeria is the location with the greatest proportion of its elements in this category, and Hong Kong has the fewest.

> Recommendation: Consider including the history of computing.

Part of the reason for Nigeria's high percentage is that one of the elements in the Impacts of Computing category is "explore the history of computing," with five instances, all from Nigeria. These elements cover topics ranging from the history of calculating devices (Nigeria, N3.1) to comparing ancient and modern methods for transmitting information (Nigeria, NJ1.5).

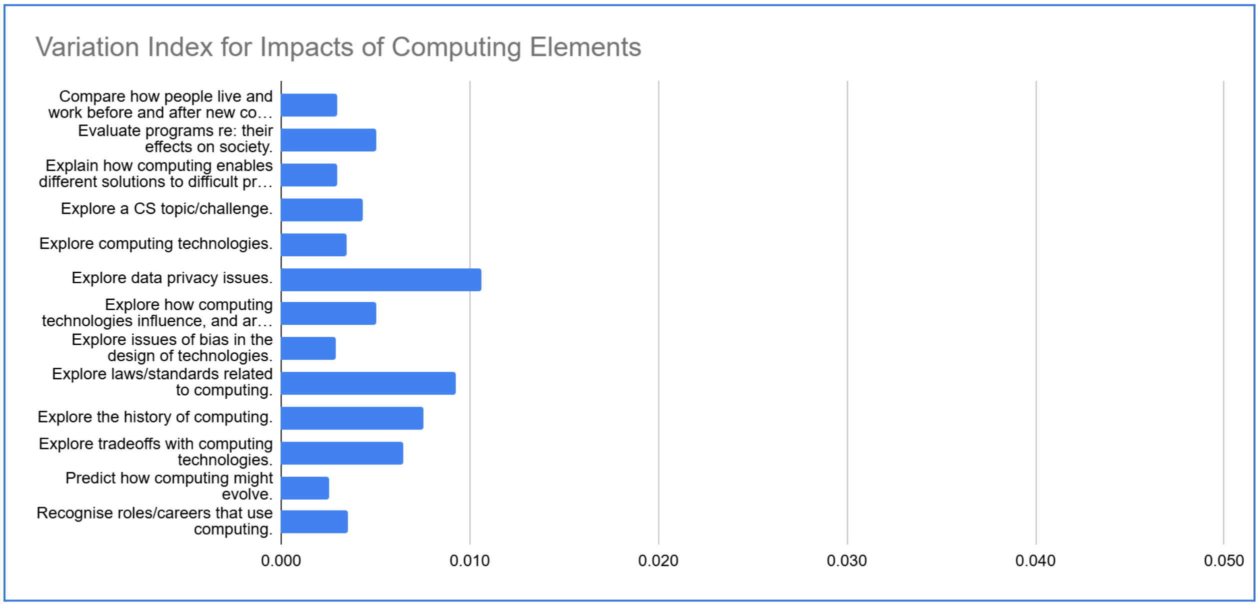


Recommendation: Consider how much to emphasize laws/standards relative to other societal impacts.

As the pie charts (see above) show, the element "explore laws/standards related to computing" is much more common in the US than elsewhere, spanning all grade bands.

Consider standards that explore the role of computing in careers.

The pie charts also show that the element "recognise roles/careers that use computing" is not common across locations.



Recommendation: Consider increasing the emphasis on data privacy.

As the figure above, Variation Index for Impacts of Computing, shows, the highest variation in this category is for the element "explore data privacy issues." (The elements with the lowest variations tend to be rare across locations.) This element is not very common in the US (0.6% of all elements, relative to 1.5% of all elements in all locations). Australia and Finland include explorations of data privacy at every grade band. Societal impacts related to data privacy are likely to increase due to AI, the internet of things, and similar technologies.

Recommendation: Consider including personal impacts.

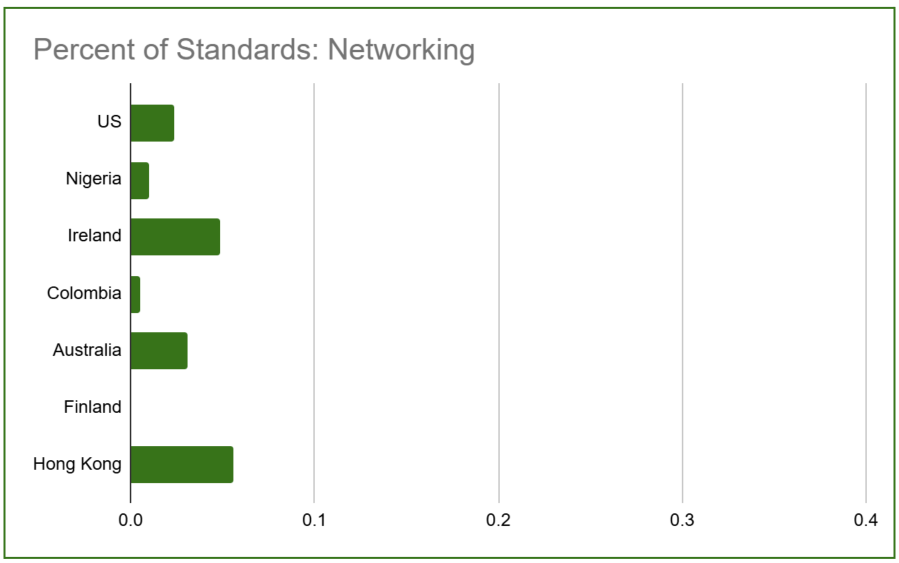
Some of the Colombian standards that contain Impacts of Computing elements emphasize the personal impact as well as societal impact, something not common in other location's standards. For example, C2.9 reads: "recognizes that spending too much time using digital technologies can affect their emotions and looks for strategies to achieve a balance between time dedicated to computers and the rest of their daily lives" and C5.14 reads: "identifies the impact of technology (for example, the use of social media, cyberbullying, cybersecurity) in their personal life and in society."

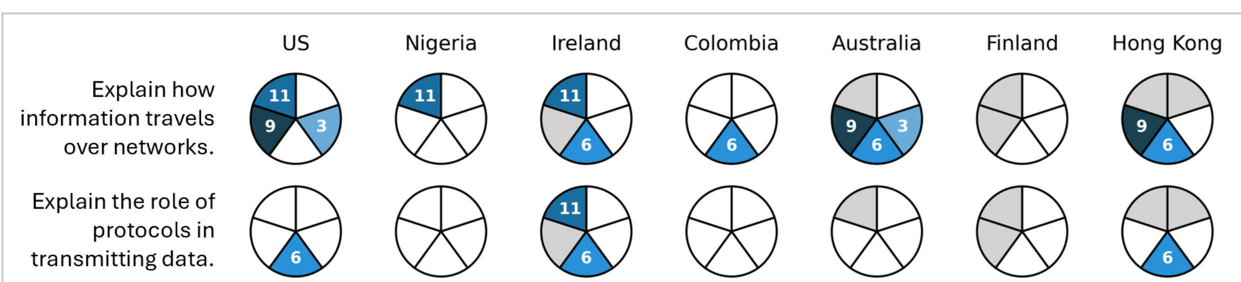
Recommendation: Consider including a CS-related research project.

The element “explore a CS topic/challenge” contains three instances, all from Ireland. They are found in three different standards, all in the Junior Cycle (≈ ages 12 - 14), and focused on identifying a topic or challenge in CS (Ireland, J2.7), researching it (Ireland, J2.8), and presenting a related proposal for discussion and feedback (Ireland, J2.9). These are the only standards (across all locations and categories) that explicitly required research into a CS-related topic of the student’s choice. (We note that, depending on the student’s choice of topics, these standards may or may not involve *impacts* of computing.)

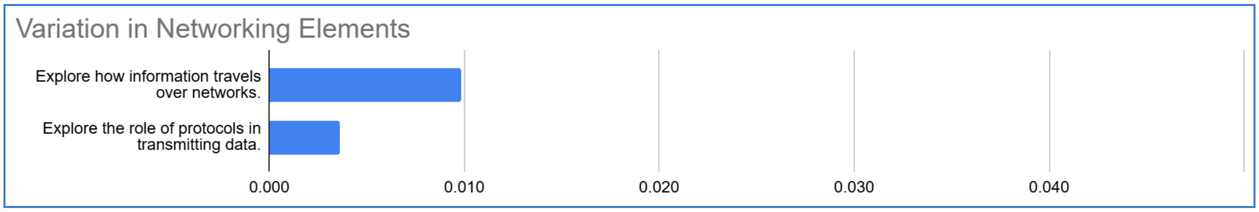
4.2.5 Networking

As Percent of Standards: Networking (see below) indicates, Networking is not a common category in any location, constituting <6% of the elements in each location. Finland has no Networking elements, perhaps reflecting that Finland’s standards are meant to be integrated into other subject areas.



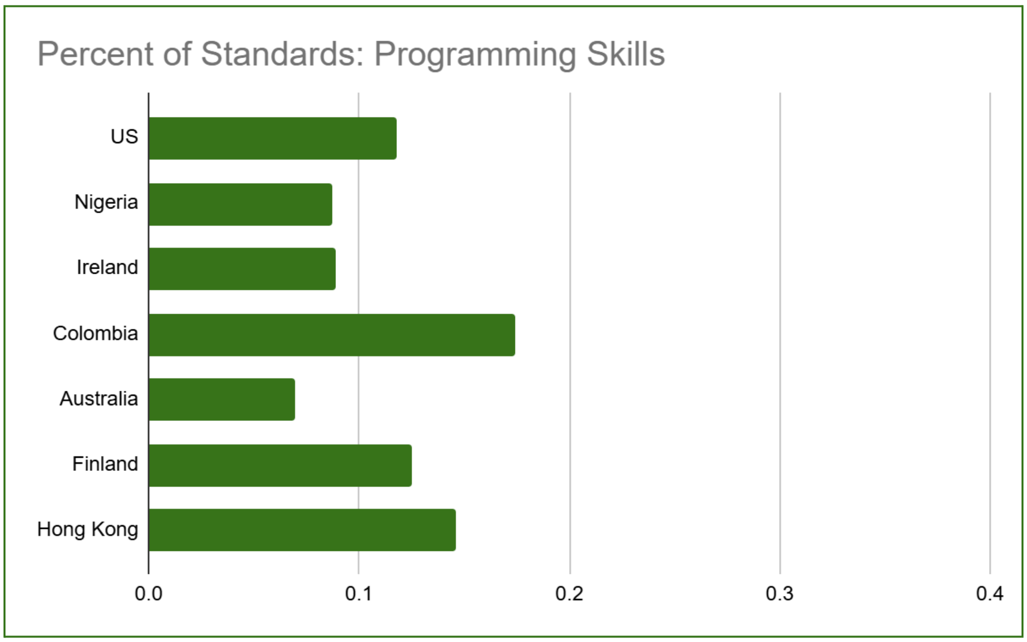


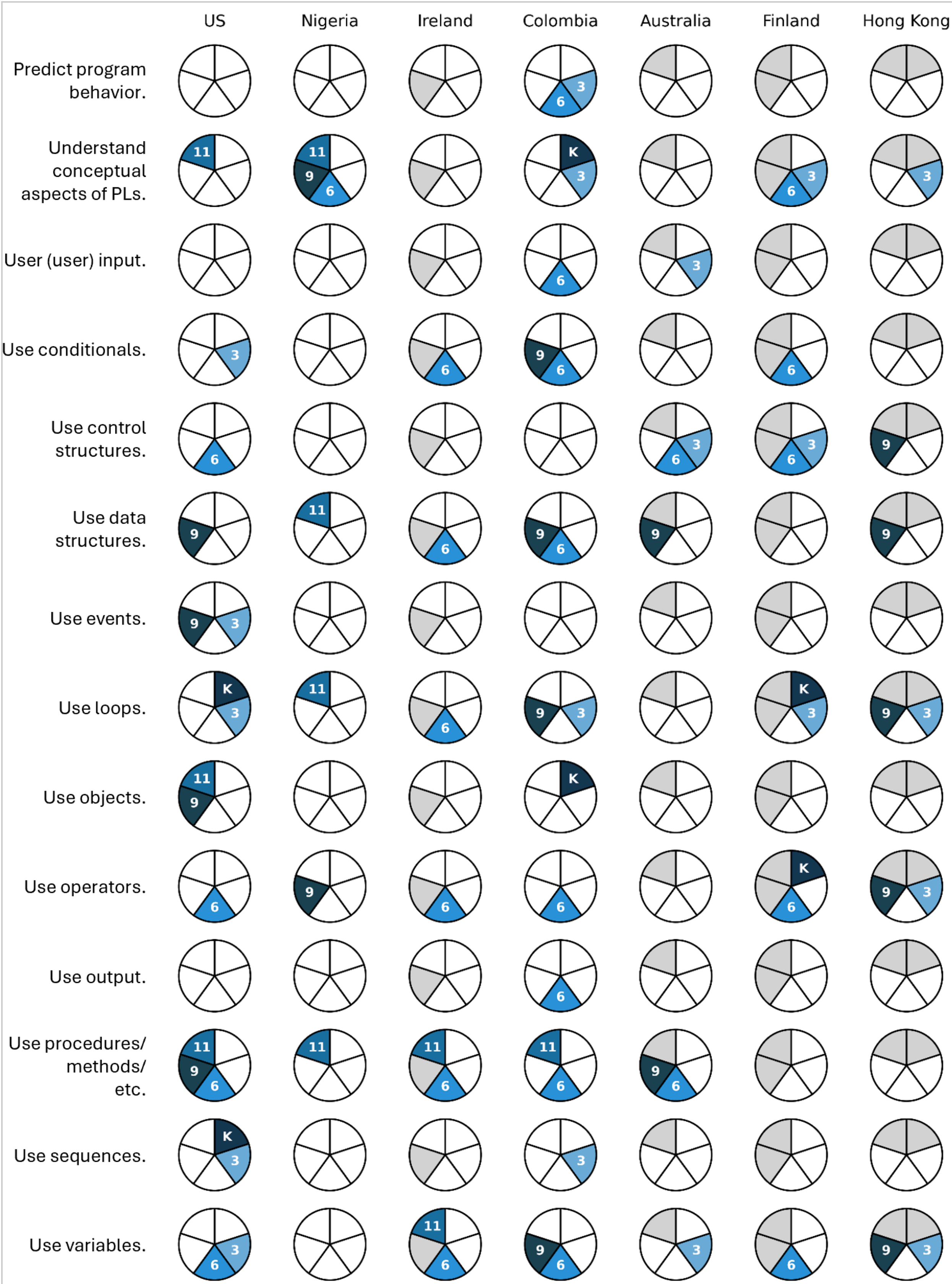
As the pie charts above show, the element “explore how information travels over networks” appears across grade bands in Australia and in the US.



4.2.6 Programming Skills

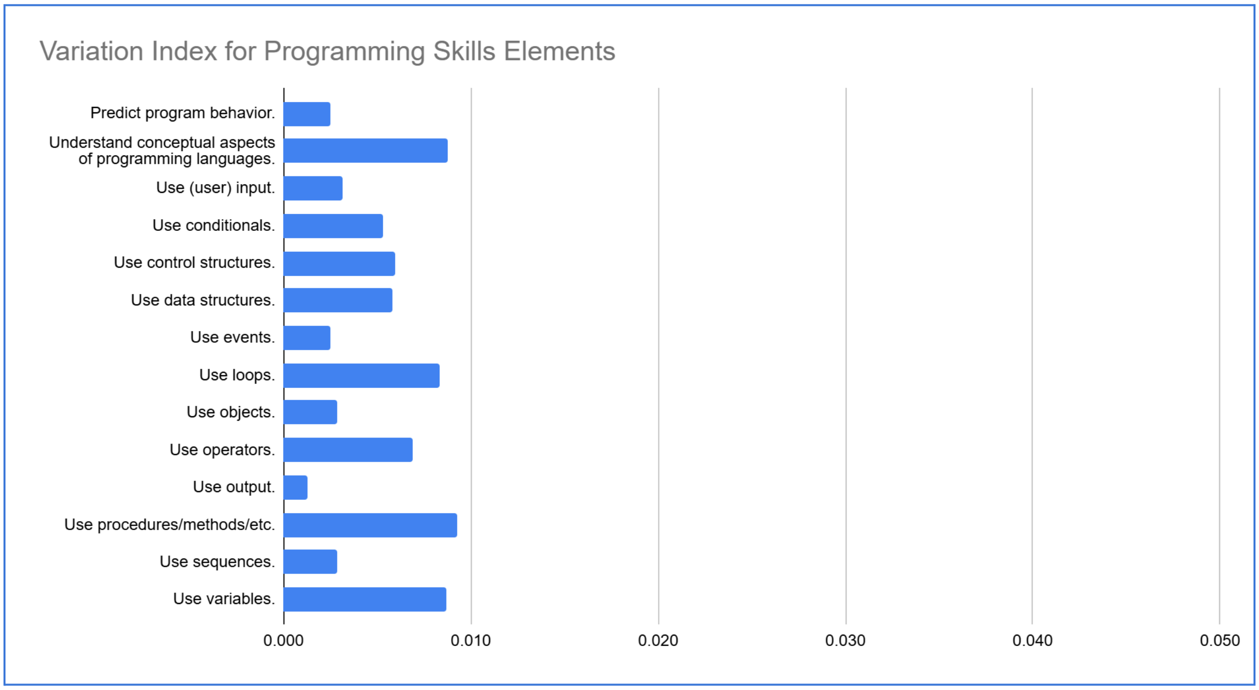
As the figure (see below) Percent of Standards: Programming Skills illustrates, this category forms the largest portion of elements in Colombia and is least common in Australia.




US
Nigeria
Ireland
Colombia
Australia
Finland
Hong Kong
Predict program behavior.
Understand conceptual aspects of PLs.
User (user) input.
Use conditionals.
Use control structures.
Use data structures.
Use events.
Use loops.
Use objects.
Use operators.
Use output.
Use procedures/ methods/ etc.
Use sequences.
Use variables.

Recommendation: Consider alignment of programming concepts across grade levels/bands.

As the pie charts suggest, many locations include Programming elements in a manner that does not appear to be well-articulated across grade bands. For example, US standards reference events in 3-5 and 9-10, but not 6-8. Similarly, Colombian standards mention loops in 3-5 and 9-10, but not 6-8. In both cases, students may retain the programming concepts better without this gap.



As the figure above, Variation Index for Programming Skills Elements, illustrates, there is a higher level of variation for elements involving conceptual elements, loops, operators, procedures, and variables relative to other elements. In general, there appear to be few patterns concerning the alignment, count/percent, and scope/sequence of programming elements in the various locations.

The element "understand conceptual aspects of programming languages" includes a variety of concepts, ranging from comparing multiple programming languages (US, 3B-AP-24) to conceptualizing a programming language as a formal set of instructions (Colombia, C2.4) to distinguishing high-level from other programming languages (Nigeria, NS3.7).

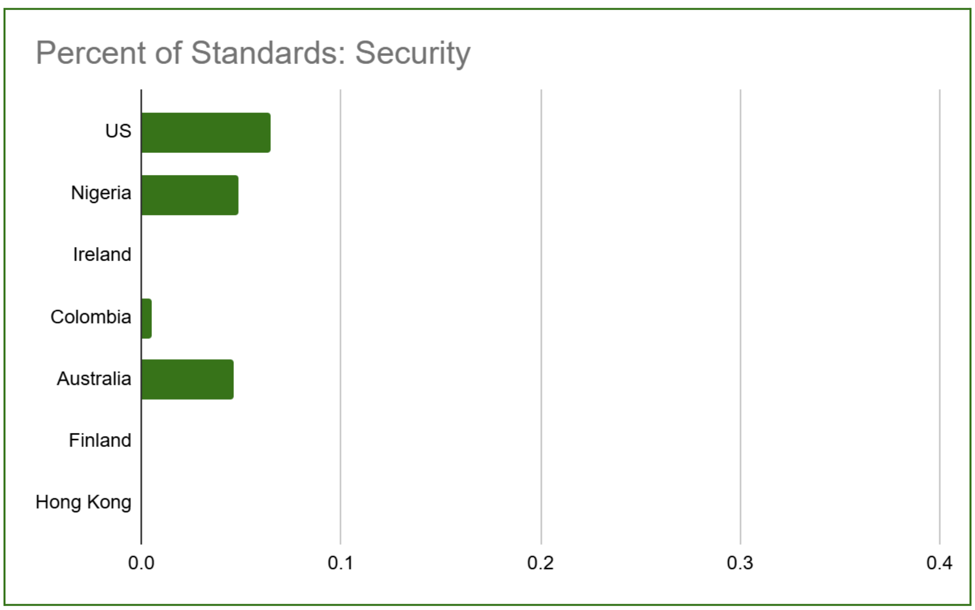
Nigeria is unusual in that its standards mention a specific general-purpose programming language, BASIC (e.g., Nigeria, NS1.10). In other locations, program language specification is normally limited to references to block-based or text-based languages and not specific languages per se. While Nigeria's choice provides clarity and specificity, it also makes the standards less flexible and subject to being more quickly dated. (Note that classrooms in Nigeria use other languages, such as Scratch and Python, but these languages are not mentioned in the standards themselves.)

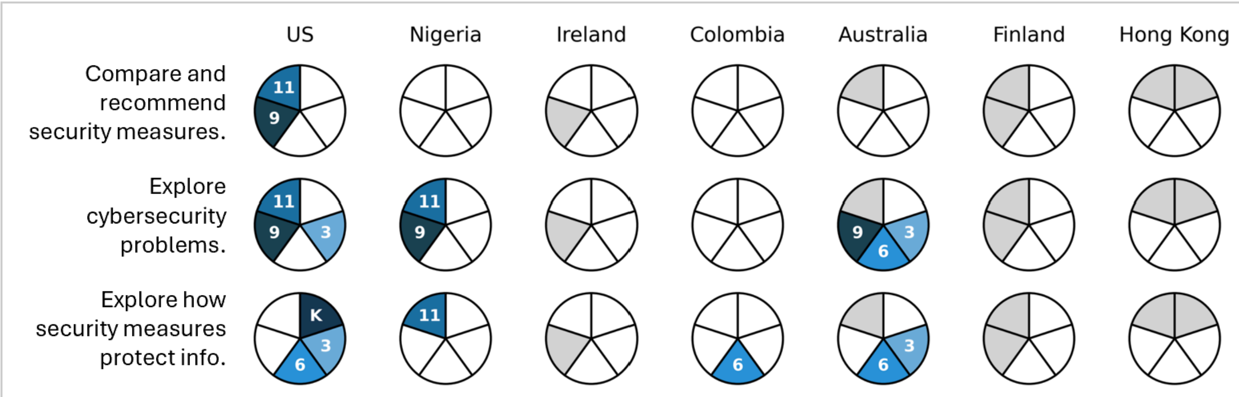
Recommendation: Consider including predicting program behavior.

The element "predict program behavior" has two instances, both in Colombia (Colombia, C3.6 and Colombia, C5.6). Predicting programming behavior is an important skill. As part of a structured approach to learning programming (e.g., PRIMM, or predict, run, investigate, modify, make), it has been shown to improve learning outcomes and support teacher effectiveness (Sentance et al., 2019). This skill may take on added importance when working with AI tools. While predicting programming behavior may be an implicit skill for some extant standards, articulating it explicitly may encourage greater focus on it and ensure that it is not overlooked.

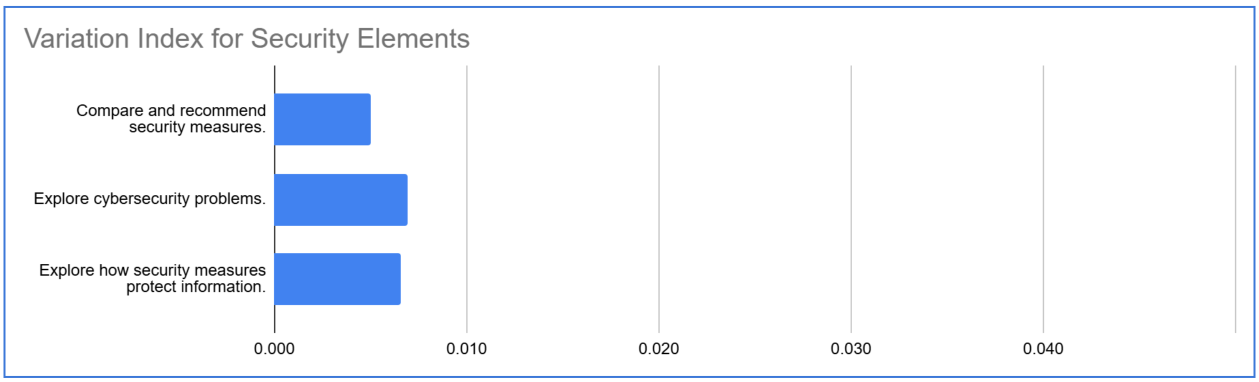
4.2.7 Security

As the figure below (Percent of Standards: Security) shows, this category is relatively rare in all locations, and no elements from it occur in Ireland, Finland, or Hong Kong. Given the importance of secure programming – as well as the economic, personal, and political issues security breaches can present – this is perhaps somewhat surprising.



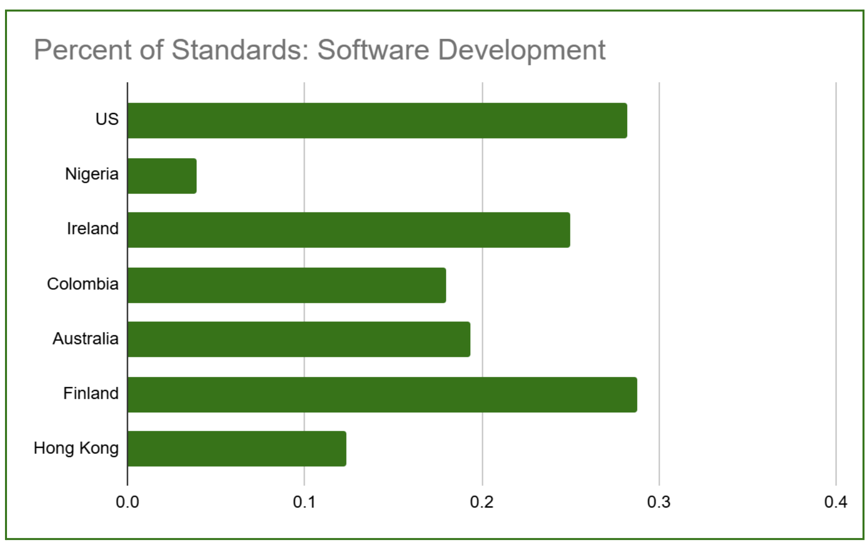


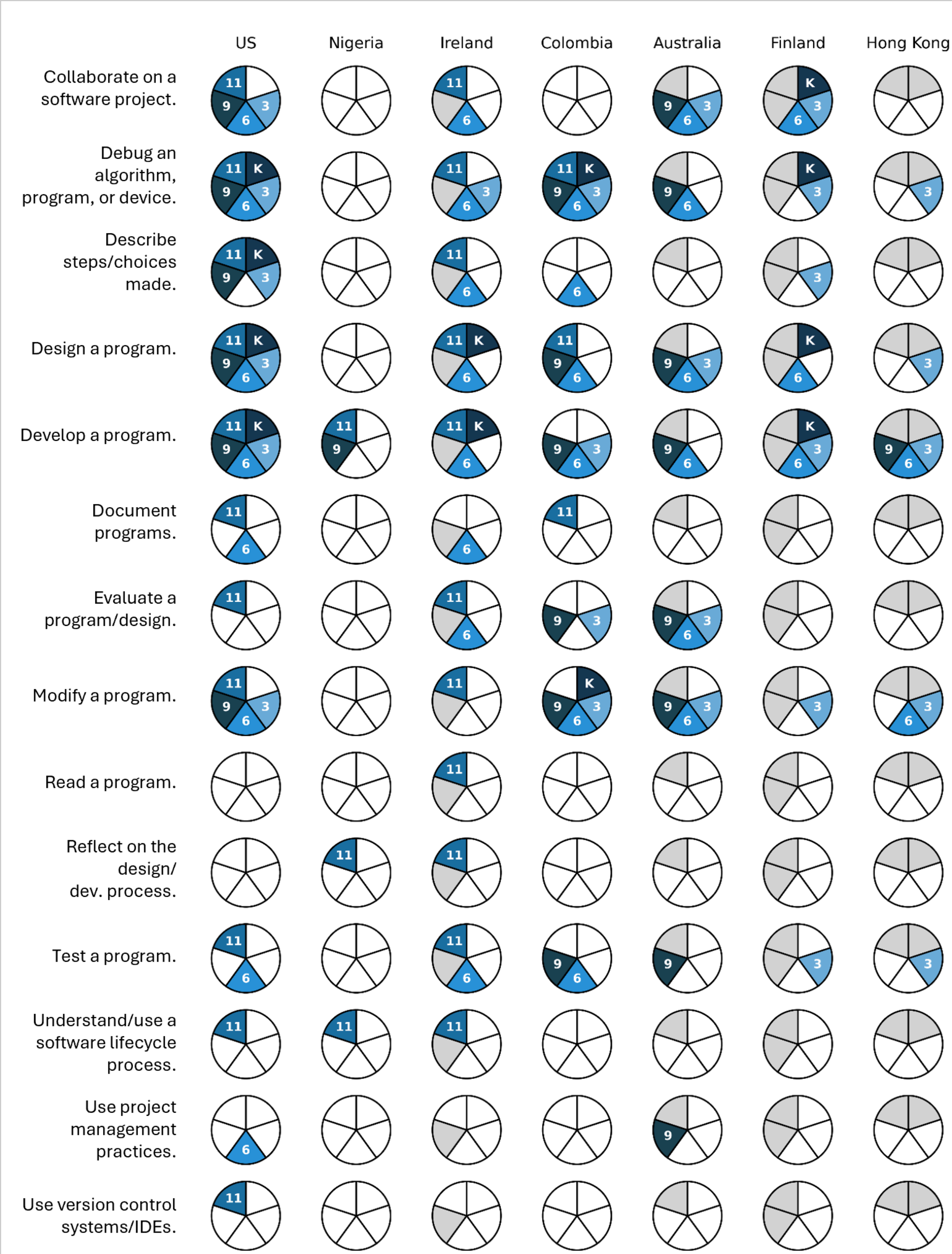
The pie charts above show that security elements generally tend to span grade bands in the US and Australia but occur in the higher bands in Nigeria and are largely absent elsewhere.



4.2.8 Software Development

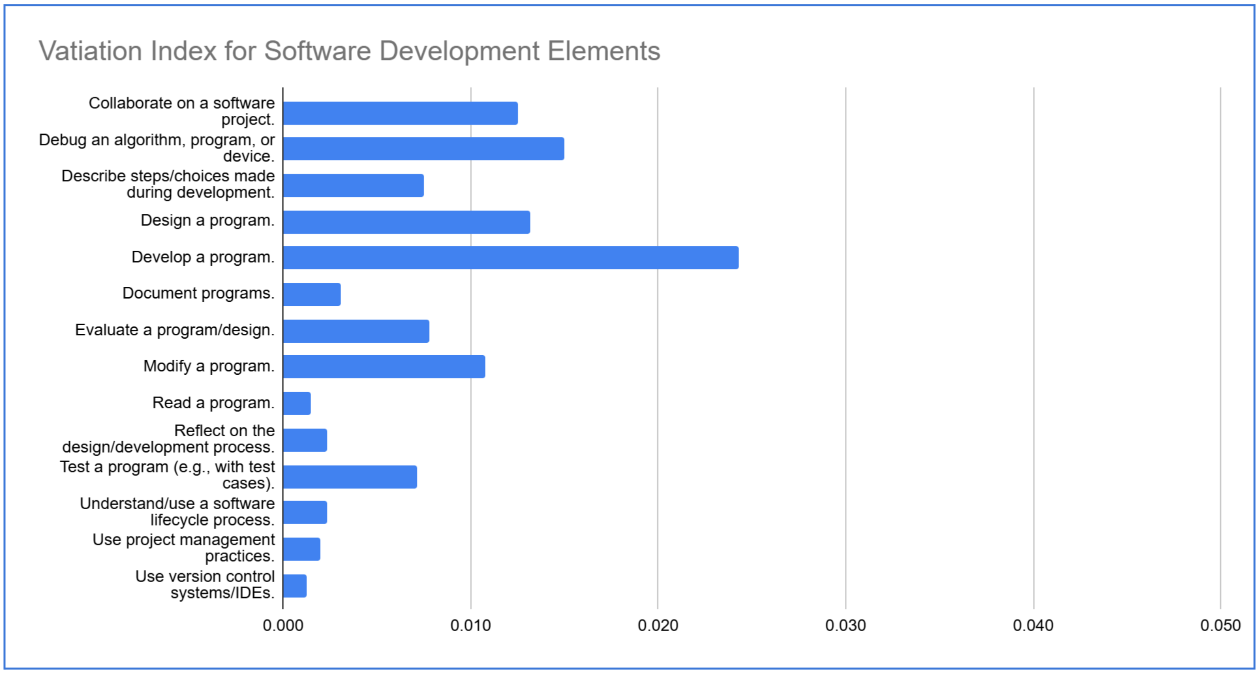
The figure Percent of Standards: Software Development shows that, for some locations (the US and Finland, Software Development elements compose over one-quarter of their elements, while in other locations (especially Nigeria), this category is much less common. The most common of the Software Development elements is “develop a program,” with 42 instances.




US
Nigeria
Ireland
Colombia
Australia
Finland
Hong Kong
Collaborate on a software project.
Debug an algorithm, program, or device.
Describe steps/choices made.
Design a program.
Develop a program.
Document programs.
Evaluate a program/design.
Modify a program.
Read a program.
Reflect on the design/ dev. process.
Test a program.
Understand/use a software lifecycle process.
Use project management practices.
Use version control systems/IDEs.

As the pie charts on page 34 show, Nigeria only includes software development elements at the highest grade band, while locations such as the US and Colombia include them across all grade bands.

As the figure Variation Index for Software Development Elements (see below) shows, “develop a program” has the highest variation.



Recommendation: Consider including code reading.

Only one instance of the “read a program” element exists: S1.22, in Ireland’s Senior Cycle (≈ 11 - 12 grade). Given the increased use of AI tools capable of generating (often imperfect) code, the importance of the ability to read code well is likely to increase in the coming years (Becker et al., 2023).

The element “develop a program” is one of the most common elements, with 42 instances. This element includes standards that ask students to develop specific programs as well as generic requirements to develop *any* program. This is true even with elements from the same location. For example, Colombia C8.1 reads: “creates simple programs in block and text-based languages using functions that already exist,” while Colombia C6.15 reads: “trains an image recognition application.” To better understand the diversity that exists within the “develop a program” element, we categorized each instance of this element according to whether it included each of these four features:

1. Reference to specific programming constructs (e.g., “creates simple programs in block and text-based languages *using functions* that already exist”, Colombia, C8.1)

2. Reference to software development practices (e.g., “plan and develop programs for broad audiences *using a software life cycle process*,” US, 3B-AP-17)
3. An abstract purpose for the activity (e.g., “develop programs with sequences and simple loops, *to express ideas or address a problem*,” US, 1A-AP-10)
4. A concrete description of the behavior of the program (e.g., “*generate and print a set of integers* using [a] random number generator to observe randomness of the pattern” Hong Kong, H73)

Some standards with the “develop a program” element had more than one of these features, as the table below shows:

Features				**Count**
Programming Construct	Development Practice	Abstract Purpose	Program Description	
✔				14
	✔			3
		✔		8
			✔	10
✔	✔			1
✔		✔		2
	✔	✔		2
	✔		✔	1
		✔	✔	1

Recommendation: Consider including standards that invite reflection.

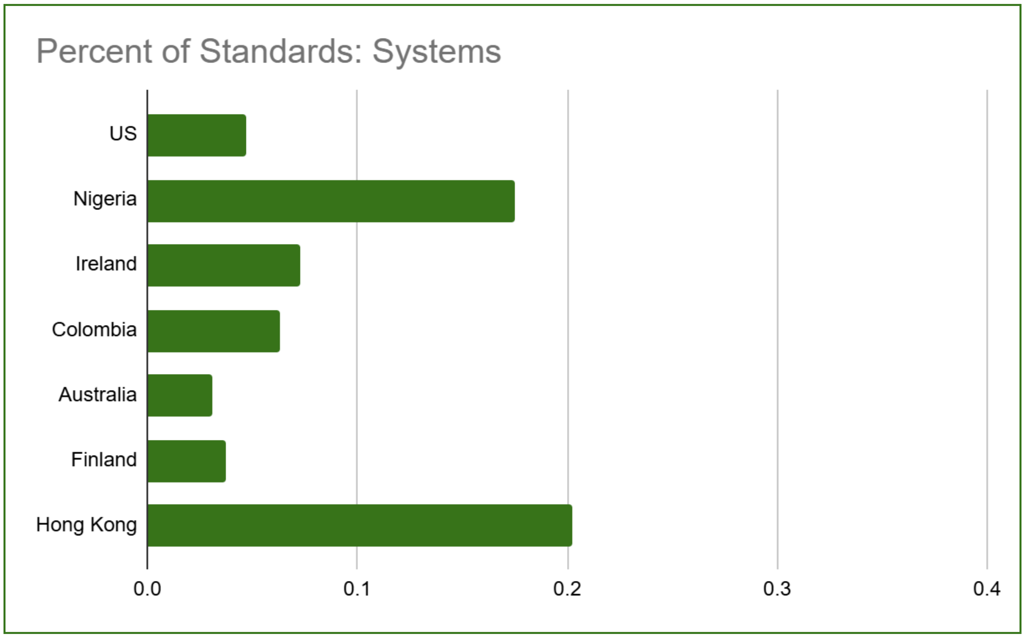
The element “reflect on the design/development process” had three instances, two focused on the design/development process and one focused on program quality:

- “describes the design process and development of a program that includes conditionals and loops and reflects about opportunities for improvement” (Colombia, C6.8)
- “. . . state the characteristics of a good program” (Nigeria, NS2.10)
- “reflect and communicate on the design and development process” (Ireland, S1.23)

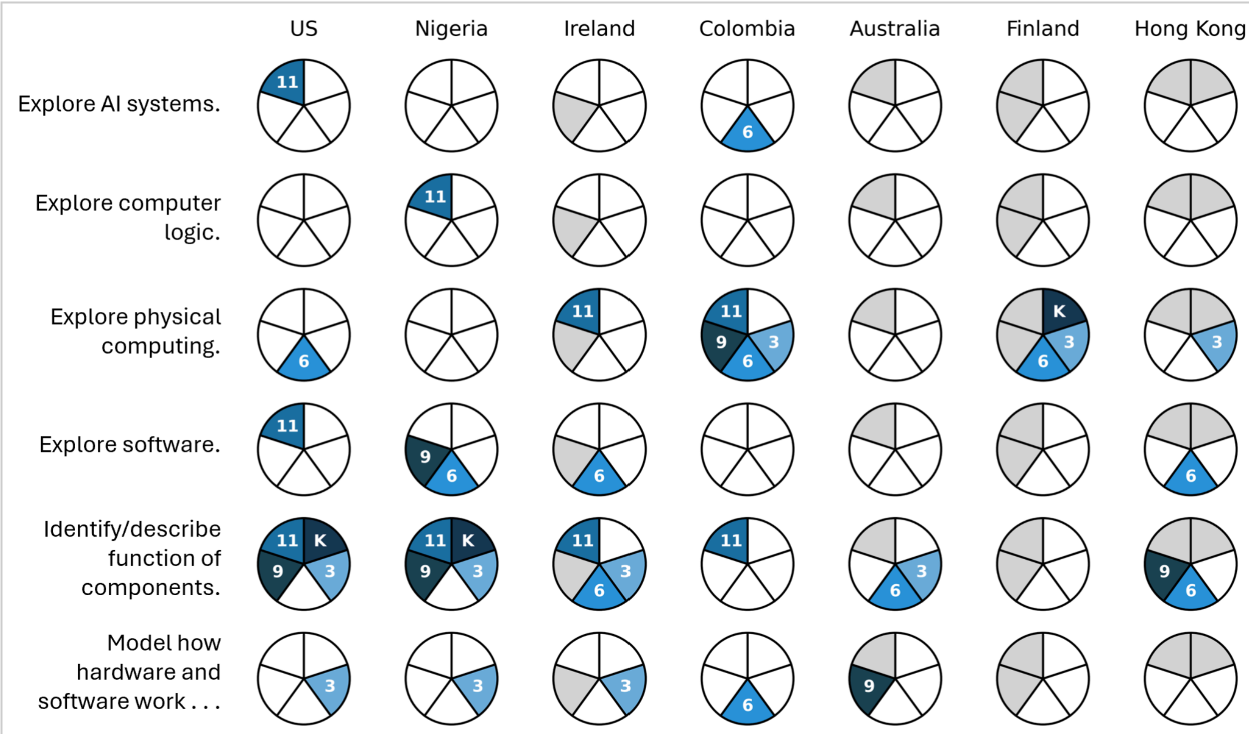
These opportunities for reflection may help students develop a disposition of reflectiveness, one of the dispositions identified by the *Reimagining CS Pathways* project as an essential aspect of a foundational CS learning experience (CSTA et al., 2024)

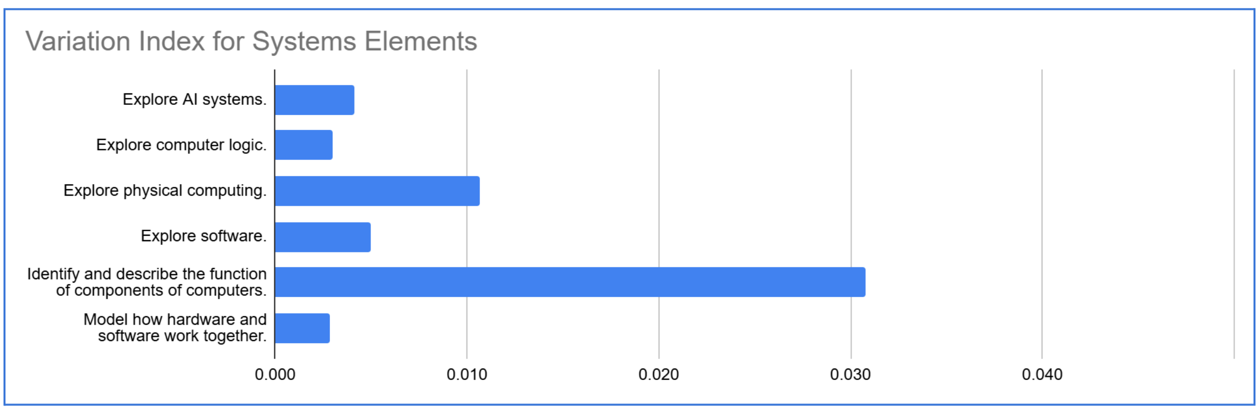
4.2.9 Systems

As the figure Percent of Standards: Systems (see below) illustrates, Hong Kong and Nigeria have the greatest proportion of their elements in this category, with far fewer in the US, Australia, and Finland.



As the pie charts on page 38 illustrate, Colombia and Finland feature the element "explore physical computing" across grade bands, something that is far less common in other locations. In contrast, "identify and describe the function of components of computers" is common across grade bands everywhere except for Colombia and Finland (where it is not included at all).

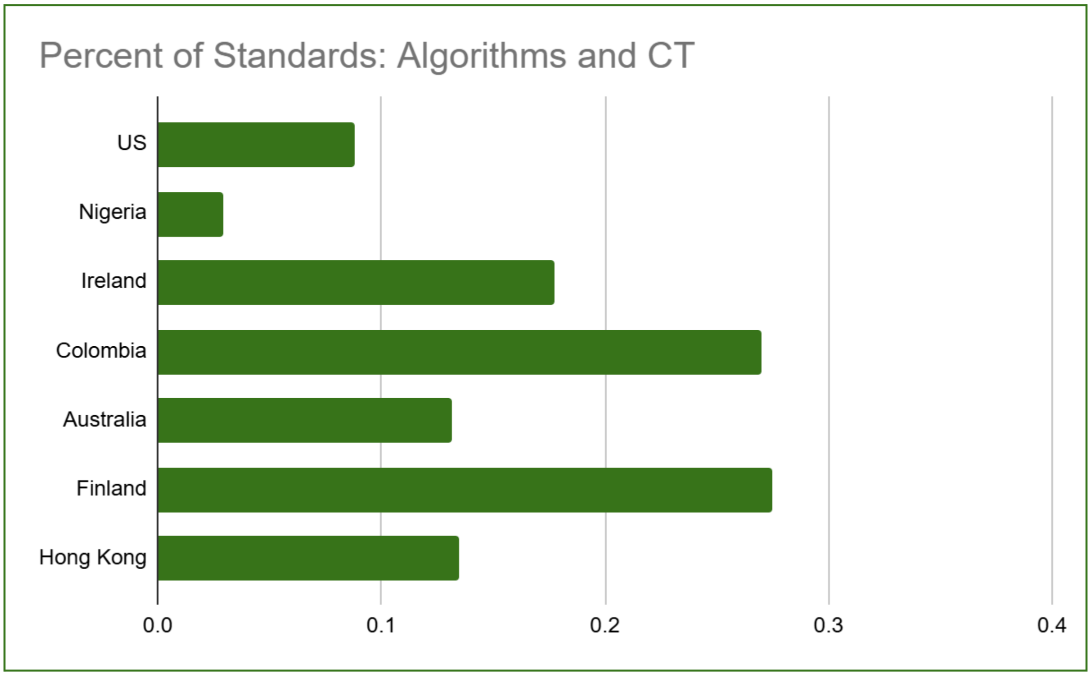


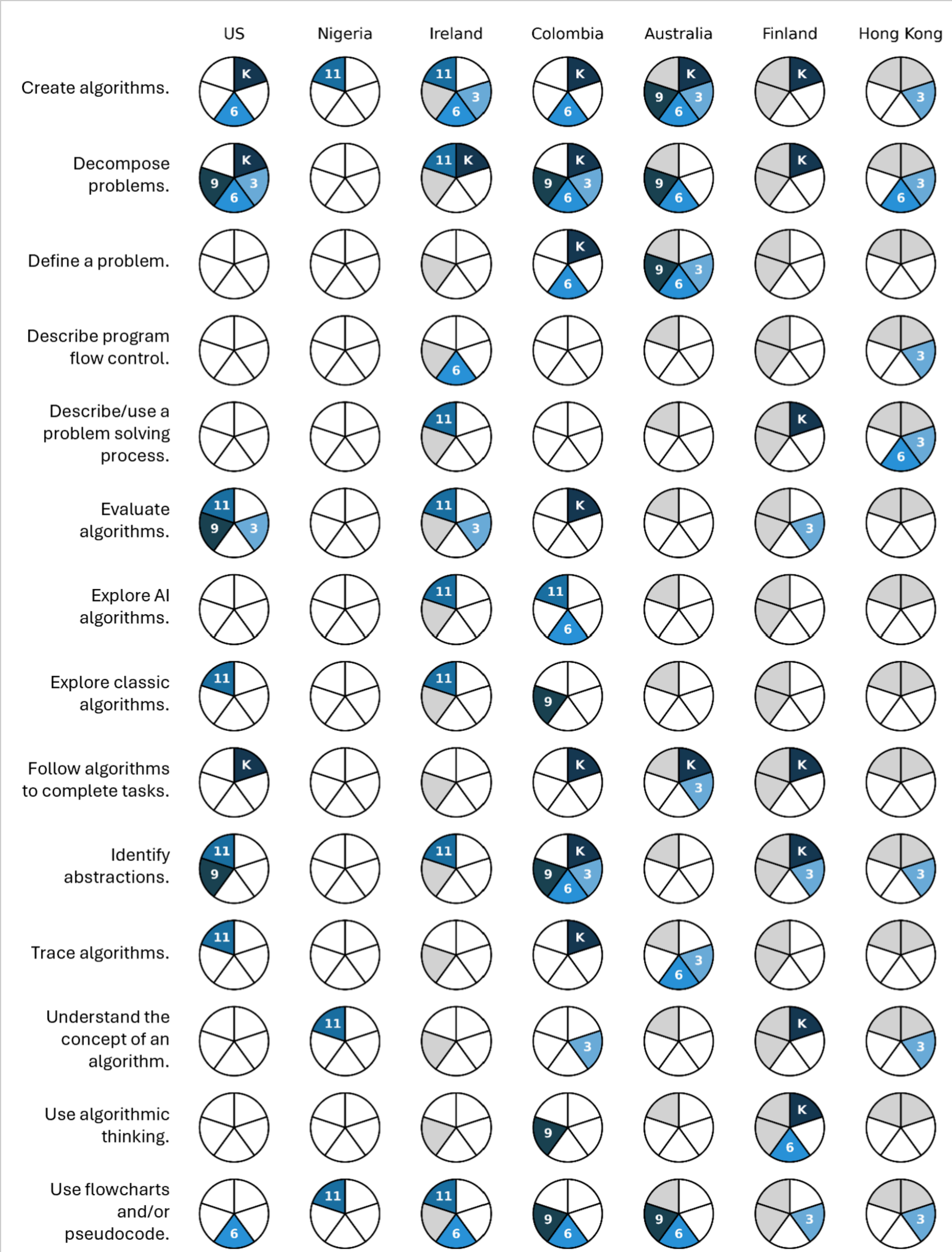


As the figure above, Variation Index for Systems Elements, shows, the variation for “identify and describe the function of components of computers” has the highest variation across locations.

4.2.10 Algorithms and Computational Thinking

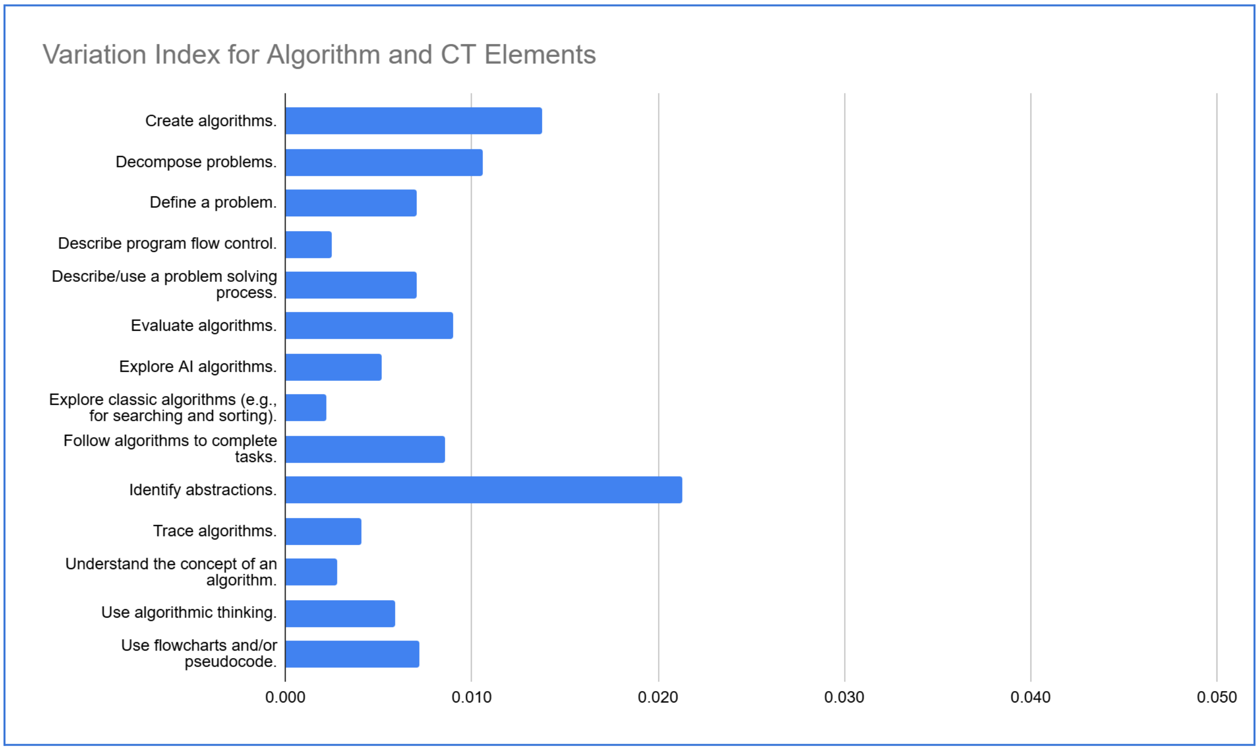
As the chart Percent of Standards: Algorithms and CT shows, Colombia and Finland emphasize these topics, with over a quarter of their elements belonging to this category. In contrast, <3% of Nigeria's elements fit into this category.




US
Nigeria
Ireland
Colombia
Australia
Finland
Hong Kong
Create algorithms.
Decompose problems.
Define a problem.
Describe program flow control.
Describe/use a problem solving process.
Evaluate algorithms.
Explore AI algorithms.
Explore classic algorithms.
Follow algorithms to complete tasks.
Identify abstractions.
Trace algorithms.
Understand the concept of an algorithm.
Use algorithmic thinking.
Use flowcharts and/or pseudocode.

As the chart on page 40 shows, there is no consensus on the placement of computational thinking topics at various grade bands. For example: the element “identify abstractions” occurs across most/all grade bands (Colombia), only at lower bands (Finland, Hong Kong), and only at higher bands (the US). Similarly, “decompose a problem” occurs across most/all grade bands (the US, Colombia), only at lower bands (Finland, Hong Kong), and only at higher bands (Ireland), and note that these band locations are not consistent across locations (e.g., the US has decomposition at most/all bands and abstractions only at lower bands). Note also that Colombia includes 14 instances of the “identify abstractions” element, making this element very common there.

As the figure below, Variation Index for Algorithm and CT Elements, illustrates, the “identify abstraction” element has the highest variation, likely due to its high occurrence in Colombia.



Throughout all content, but especially with content categorized as Algorithms and CT, Finland’s standards were quite distinct in their emphasis on play and playfulness. (Similarly, Ireland’s standards for primary CS are all introduced with the phrase “through appropriately playful and engaging learning experiences.”) Examples from Finland include:

- “Children familiarise themselves with the concept of algorithm by examining different instructions and ways to give instructions. They try giving and following instructions playfully.” (Finland, F18, Pre-Primary or age 6)

- “Children play different games and explore their properties together. Familiar play and existing games are modified with the children by inventing new rules, tasks or ways to proceed.” (Finland, F30, Early Childhood Education and Care or ages 4-5)
- “Children gain experience of creative activity and expression through technology and carry out playful tasks by using different models and instructions, as well as equipment and devices controlled with commands.” (Finland, F18, Pre-Primary or age 6)

Recommendation: Ensure that standards, especially for younger children, are developmentally appropriate.

This phenomenon may be at least partially due to the fact that Finland has standards for the youngest children, Early Childhood Education and Care, for students 4-5 years old. However, games are also emphasized across grade bands as ways to engage with computing, such as with Finland F38, a grades 7-9 standard: “is able to design and create a game, simulation or application that solves some kind of problem related to school or the student’s personal life.”

5. Conclusion

This report explored the landscape of standards across multiple locations. It is important to recognize that a student's classroom experience is shaped not by standards per se but by how those standards are implemented. While there is limited research on standards themselves, there is even less on how those standards are implemented, or the enacted curriculum. For example, this report notes that Nigeria's standards emphasize digital literacy. But a recent survey of Nigerian computer science teachers found that many topics that are either not included or not emphasized in Nigeria's CS standards are covered by some classes, including robotics (20% of teachers), machine learning (20%), and design process (27%) (Tshukudu et al., 2023). Similarly, in Australia, robotics – which is, unlike in other locations, not specified in the standards – is covered by 79% of CS teachers, and the topic of data analysis and visualization – which *is* specified – is covered by only 29% of CS teachers.[2] Therefore, in addition to the recommendations included throughout this report which are geared toward standards writers, we offer an additional recommendation for the computer science education research community: *Add to the research base on the enacted curriculum.* A better understanding of this broader landscape of CS education will provide more insight into how standards are implemented at the classroom level.

The body of international standards explored in this report presents a wide array of knowledge, skills, and dispositions – with broad commonalities across locations, but also with distinctive elements in each location. We hope that the analysis presented in this report will equip standards writers to thoughtfully borrow from their colleagues in other locations, enabling efforts to improve computer science education for students worldwide.

[2] 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.

Appendices

Appendix A: Standard Organization by Location

US (CSTA)

Taxonomy		Level				
Concept	Subconcept	K - 2	3 - 5	6 - 8	9 - 10	11 - 12
Computing Systems	Devices	✔	✔	✔	✔	
	Hardware & Software	✔	✔	✔	✔	✔
	Troubleshooting	✔	✔	✔	✔	
Networks & the Internet	Network Communication & Organization		✔	✔	✔	✔
	Cybersecurity	✔	✔	✔	✔	✔
Data & Analysis	Storage	✔	✔	✔	✔	
	Collection, Visualization, & Transformation	✔	✔	✔	✔	✔
	Inference & Models	✔	✔	✔	✔	✔
Algorithms & Programming	Algorithms	✔	✔	✔	✔	✔
	Variables	✔	✔	✔	✔	✔
	Control	✔	✔	✔	✔	✔
	Modularity	✔	✔	✔	✔	✔
	Program Development	✔	✔	✔	✔	✔
Impacts of Computing	Culture	✔	✔	✔	✔	✔
	Social Interactions	✔	✔	✔	✔	
	Safety, Law, & Ethics	✔	✔	✔	✔	✔

Nigeria

Taxonomy		Level											
		Primary						Junior Secondary			Senior Secondary		
Subject	Theme	1	2	3	4	5	6	1	2	3	1	2	3
Information Technology	Basic Computer Operations	✔	✔	✔	✔	✔	✔	✔	✔	✔			
	Basic Concepts of IT	✔	✔	✔	✔	✔	✔	✔	✔	✔			
	Computer Application Packages								✔	✔			
Computers & IT	Fundamentals of Computers										✔		
	Computer Evolution										✔		
	Basic Concepts of Computer Hardware										✔	✔	
	Computer Hardware										✔	✔	
	Basic Concepts of Computer Software										✔		
	Developing Problem-Solving Skills										✔	✔	✔
	Information and Communications Technology										✔	✔	✔
	Operating the Computer										✔	✔	
	Computer Applications										✔	✔	✔
	Handling Computer Files											✔	
	Coding Systems in Computer[s]												✔
	Computer Ethics												✔

Ireland

Taxonomy		Level					
Strand	Substrand	Stage 1	Stage 2	Stage 3	Stage 4	Junior Cycle	Senior Cycle
Technology		✔	✔	✔	✔		
Computer Science Introduction	My Digital World					✔	
	Being a Coder -- Step by Step					✔	
Let's Get Connected	Making Connections					✔	
	Bits and Bytes					✔	
	Real World Problems					✔	
Coding at the Next Level	Being a Coder					✔	
	Documenting the Code					✔	
Principles and Practices	Computational Thinking						✔
	Computers and Society						✔
	Designing and Developing						✔
Core Concepts	Abstraction						✔
	Algorithms						✔
	Computer Systems						✔
	Data						✔
	Evaluation and Testing						✔
Computer Science in Practice	Interactive Information Systems						✔
	Analytics						✔
	Modelling and Simulation						✔
	Embedded Systems						✔

Colombia

Taxonomy		Level											
Category	Subcategory	0	1	2	3	4	5	6	7	8	9	10	11
Concepts and Abilities	Algorithms and Programming	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
	Patterns and Decomposition	✔	✔			✔	✔	✔	✔	✔		✔	
	Data	✔	✔	✔	✔	✔		✔	✔	✔	✔	✔	✔
Digital Citizenship	Impacts of Digital Technology		✔	✔			✔	✔	✔	✔	✔	✔	✔
	Security in the Digital World				✔					✔			
Problem Solving Practice Using Computing	Physical Computing					✔	✔		✔	✔		✔	✔
	Management of Data and Information						✔	✔	✔		✔	✔	✔
	Artificial Intelligence						✔	✔	✔	✔			✔
	Modeling and Simulation							✔	✔		✔	✔	✔

Australia

Note that content related to Data Representation is included in the table below per Australia's organization, but the content itself was located in Mathematics (Statistics) for logistical reasons and therefore is not be fully included in this project.

Taxonomy		Level					
Strand	Substrand	Foundation	Years 1 - 2	Years 3 - 4	Years 5 - 6	Years 7 - 8	Years 9 -10
Knowledge and Understanding	Digital Systems	✔	✔	✔	✔	✔	✔
	Data Representation	✔	✔	✔	✔	✔	✔
Processes and Production Skills	Acquiring, Managing, and Analysing Data					✔	✔
	Investigating and Defining		✔	✔	✔	✔	✔
	Generating and Designing		✔	✔	✔	✔	✔
	Producing and Implementing			✔	✔	✔	✔
	Evaluating		✔	✔	✔	✔	✔
	Collaborating and Managing		✔	✔	✔	✔	✔
	Privacy and Security	✔	✔	✔	✔	✔	✔

Finland

Taxonomy: Area	Taxonomy: Theme	Level: ECE	Pre-	1-2	3-6	7-9
Computational Thinking	Logical Thinking and Processing of Information	✔	✔	✔	✔	✔
	Solving and Modelling of Problems	✔	✔	✔	✔	✔
	Activities, Concepts, and Basic Structures of Programming	✔	✔	✔	✔	✔
Inquiry-Based Work and Producing	Co-Creation Processes	✔	✔	✔	✔	✔
	Creative Production	✔	✔	✔	✔	✔
	Programming as a Tool for Learning	✔	✔	✔	✔	✔
	Practical Skills	✔	✔	✔	✔	✔
Programmed Environments and Acting in Them	Programmed Technology in Different Areas of Life	✔	✔	✔	✔	✔
	Impacts of Programmed Technology in Everyday Life	✔	✔	✔	✔	✔

Hong Kong

Taxonomy: Knowledge	Taxonomy: Learning Element	Level: Stage 2	Secondary 1	Secondary 2	Secondary 3
Computational Thinking		✔			
Information and Communication Technology	Computer System		✔		
	Information Processing and Presentation		✔	✔	✔
	Programming Concepts			✔	✔
	Computer Networks			✔	✔

Becker, B. A., Denny, P., Finnie-Ansley, J., Luxton-Reilly, A., Prather, J., & Santos, E. A. (2023). Programming Is Hard - Or at Least It Used to Be: Educational Opportunities and Challenges of AI Code Generation. *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, 500–506. https://doi.org/10.1145/3545945.3569759

Bell, T., Vivian, R., & Falkner, K. (2024). *Computational Thinking Curricula in Australia and New Zealand*. https://direct.mit.edu/books/oa-edited-volume/5786/chapter/4754152/Computational-Thinking-Curricula-in-Australia-and

Chin, M. (2021, September 22). *Students who grew up with search engines might change STEM education forever*. The Verge. https://www.theverge.com/22684730/students-file-folder-directory-structure-education-gen-z

CSTA, IACE, ACM, Code.org, College Board, CSforAll, & ECEP Alliance. (2024). *Reimagining CS Pathways: Every student prepared for a world powered by computing*. https://reimaginingcs.org/wp-content/uploads/2024/07/CSTA-Reimagining-CS-Pathways_v4.0.pdf

Forstag, E. H. (Ed.) (with Board on Science Education, Division of Behavioral and Social Sciences and Education, & National Academies of Sciences, Engineering, and Medicine). (2023). *Foundations of Data Science for Students in Grades K-12: Proceedings of a Workshop*. National Academies Press. https://doi.org/10.17226/26852

Kong, S.-C., & Kwok, W.-Y. (2024). Computational Thinking Education in Hong Kong. In H. Abelson & S.-C. Kong (Eds.), *Computational Thinking Curricula in K–12* (pp. 33–54). The MIT Press. https://doi.org/10.7551/mitpress/14041.003.0005

Mäkitalo, K., Tedre, M., Laru, J., Valtonen, T., Iwata, M., & Koivisto, J. (2024). *Learning Computational Thinking in Co-Creation Projects with Modern Educational Technology: Experiences from Finland*. https://direct.mit.edu/books/oa-edited-volume/5786/chapter/4754162/Learning-Computational-Thinking-in-Co-Creation

Malmi, L., Hellas, A., Ihantola, P., Isomöttönen, V., Jormanainen, I., Kilamo, T., Knutas, A., Korhonen, A., Laakso, M.-J., López-Pernas, S., Poranen, T., Salakoski, T., & Suhonen, J. (2023). Computing Education Research in Finland. In M. Apiola, S. López-Pernas, & M. Saqr (Eds.), *Past, Present and Future of Computing Education Research* (pp. 335–372). Springer International Publishing. https://doi.org/10.1007/978-3-031-25336-2_16

Niemelä, P., Pears, A., Dagienė, V., & Laanpere, M. (2022). Computational Thinking – Forces Shaping Curriculum and Policy in Finland, Sweden and the Baltic Countries. In D. Passey, D. Leahy, L. Williams, J. Holvikivi, & M. Ruohonen (Eds.), *Digital Transformation of Education and Learning—Past, Present and Future* (Vol. 642, pp. 131–143). Springer International Publishing. https://doi.org/10.1007/978-3-030-97986-7_11

Sentance, S., Waite ,Jane, & and Kallia, M. (2019). Teaching computer programming with PRIMM: A sociocultural perspective. *Computer Science Education*, *29*(2–3), 136–176. https://doi.org/10.1080/08993408.2019.1608781

Tshukudu, E., Sentance, S., Adelakun-Adeyemo, O., Nyaringita, B., Quille, K., & Zhong, Z. (2023). Investigating K-12 Computing Education in Four African Countries (Botswana, Kenya, Nigeria, and Uganda). *ACM Transactions on Computing Education*, *23*(1), 9:1-9:29. https://doi.org/10.1145/3554924

Webb, N. M. (2017). Peer instruction, problem solving, and cognition: Multidisciplinary perspectives. *International Journal of Educational Research*, *13*, 21–39.

Appendix C: Calculating the Variation Index

We wanted to be able to express numerically the extent to which the count of an element varied from location to location. To do this, we use a metric that we call the variation index, which is a variant of the mean absolute deviation. We calculated it as follows (see the figure below):

1. Determine for each element what percent it is of all elements (across all locations).
2. For each element and each location, determine what percent this element is of this location’s elements.
3. Subtract the value from step #1 from the value from step #2, and then find the absolute value of the result.
4. Find the average of the values from step #3 for every location.

$$\text{absolute value}\left[\frac{\text{count of this element for this location}}{\text{count of all elements for this location}} - \frac{\text{count of this element for all locations}}{\text{count of all elements for all locations}}\right]$$

Thus, a low variation index suggests that locations are relatively similar in how common this element is in their standards. In contrast, a higher variation index suggests greater differences across locations. The variation index ranged from 0.001 to 0.047.

This appendix includes all of the recommendations found in the report. Please see the section where each recommendation appears (listed in parentheses after each recommendation) for more context for the recommendation.

1. Analyze drafts of proposed standards to ensure that coverage of categories aligns with identified priorities. (4.1.2)
2. Consider grouping standards to promote project-based learning. (4.1.2)
3. Ensure that the cognitive complexity of standards aligns with what writers have determined is most appropriate. (4.1.4)
4. Use CS tools and skills to meet learning objectives for other subjects. (4.2.1)
5. Center equity via examples, especially in supplementary materials. (4.2.1)
6. Consider whether and how Digital Literacy should be included in CS standards, other subjects, and/or integrated across subjects. (4.2.2)
7. Determine whether skills not included previously, such as understanding how file structure works, should be included in response to changes in user behavior. (4.2.2)
8. Consider including standards involving evaluating digital tools, a task adults are likely to do regularly; it also provides opportunities to explore key elements identified by the Reimagining CS Pathways project (e.g., reflectiveness, critical thinking, impacts and ethics, inclusive collaboration, and human-centered design). (4.2.3)
9. Consider including the history of computing. (4.2.4)
10. Consider how much to emphasize laws/standards relative to other societal impacts. (4.2.4)
11. Consider standards that explore the role of computing in careers. (4.2.4)
12. Consider increasing the emphasis on data privacy. (4.2.4)
13. Consider including personal impacts. (4.2.4)
14. Consider including a CS-related research project. (4.2.4)
15. Consider alignment of programming concepts across grade levels/bands. (4.2.6)
16. Consider including predicting program behavior. (4.2.6)
17. Consider including code reading. (4.2.8)
18. Consider including standards that invite reflection. (4.2.8)
19. Ensure that standards, especially for younger children, are developmentally appropriate. (4.2.10)