Introduction

The 2017 Computer Science Standards of Learning for Virginia Public Schools identify academic content for essential components of the computer science curriculum at different grade levels. Information from Computer Science Teachers Association, K-12 Computer Science Framework, College Board Advanced Placement Computer Science courses, Exploring Computer Science, were considered in identifying computer science content necessary for success for all students in postsecondary pursuits.

Standards are identified for kindergarten through grade eight, with an optional selection of electives modules at the middle school level, and a sequence of high school courses. The standards are organized into the following content strands: Computing Systems, Networks and the Internet, Cybersecurity, Data and Analysis, Algorithms and Programming, and Impacts of Computing. The Standards of Learning within each strand progress in complexity throughout the grade levels and into high school course content. While the standards are organized by strand and identified numerically, local curricula and pacing guides should determine the instructional sequence of the content.

The K - 8 standards were designed to be integrated into instruction in multiple subject areas including mathematics, science, history, English, fine arts, and career and technology courses. The middle school and high school electives are separate courses and modules, but where appropriate, connections are made to content in other disciplines. The high school standards are designed to provide flexibility in application of core ideas to various contexts.

The core practices of computer science, including collaboration, communication, and fostering an inclusive culture, describe the behaviors and ways of thinking that computationally literate students use to fully engage in today's data-rich and interconnected world. Collaborative computing is the process of performing a computational task by working in pairs and on teams, including working with individuals with diverse perspectives, skills, and personalities. Students will need to solicit and incorporate the feedback of others, which can lead to better outcomes than working independently. Students should use collaborative tools to effectively work together and to create complex artifacts. In computer science, students communicate with diverse audiences about the use and effects of computation and the appropriateness of computational choices. Students describe, justify, and document computational processes through multiple forms of media and give appropriate attribution. Clear communication includes using precise language and carefully considering possible audiences.

The use of appropriate technology and the interpretation of the results from applying technology tools must be an integral part of teaching, learning, and assessment. While some computational thinking practices can be developed with or without technology, the use of computing devices is essential for students to meet the objectives of these standards.

The Virginia Department of Education strongly encourages educators to make equitable access a guiding principle in the implementation of these computer science standards by giving all students the opportunity to learn computer science. We encourage the elimination of barriers that restrict access to computer science for students from ethnic, racial and socioeconomic groups that have traditionally been underrepresented. Schools should make every effort to ensure their

computer science classes reflect the diversity of their student population. It is only through a commitment to equitable preparation and access that true equity and excellence can be achieved.

What is Computer Science?

The Computer Science standards integrate computer literacy, educational technology, digital citizenship, information technology, and computer science. Computer literacy, educational technology, digital citizenship, and information technology are concepts that students are also exposed to in the Computer Technology Standards of Learning. In many ways, instruction in the Computer Science standards will compliment and expound upon, at a deeper level, the concepts and skills covered with the Computer Technology standards. However, there are distinct differences between computer technology and computer science. As the foundation for all computing, *computer science* is "the study of computers and algorithmic processes, including their principles, their hardware and software designs, their applications, and their impact on society" (Tucker et. al, 2006, p. 2).

Computer science builds upon the concepts of computer literacy, educational technology, digital citizenship, and information technology. The differences and relationship with computer science are described below.

- *Computer literacy* refers to the general use of computers and programs, such as productivity software. Examples include performing an Internet search and creating a digital presentation.

- *Educational technology* applies computer literacy to school subjects. For example, students in an English class can use a web-based application to collaboratively create, edit, and store an essay online.

- *Digital citizenship* refers to the appropriate and responsible use of technology, such as choosing an appropriate password and keeping it secure.

- *Information technology* often overlaps with computer science but is mainly focused on industrial applications of computer science, such as installing software rather than creating it. Information technology professionals often have a background in computer science.

What is Computational Thinking?

Also integrated throughout the Computer Science standards is the concept of computational thinking. *Computational thinking* is an approach to solving problems in a way that can be implemented with a computer. It involves the use of concepts, such as abstraction, recursion, and iteration, to process and analyze data, and to create real and virtual artifacts [Computer Science Teachers Association & Association for Computing Machinery]. Computational thinking practices such as abstraction, modeling, and decomposition connect with computer science concepts such as algorithms, automation, and data visualization. Beginning with the elementary school grades and continuing through grade 12, students should develop a foundation of computer science knowledge and learn new approaches to problem solving that captures the power of computational thinking to become both users and creators of computing technology.

Computer Science Practices for Students

The content of the Computer Science standards is intended to support the following seven practices for students: fostering an inclusive computing culture, collaborating around computing, recognizing and defining computational problems, developing and using abstractions, creating computational artifacts, testing and refining computational artifacts, and communicating about computing. The practices describe the behaviors and ways of thinking that computationally literate students use to fully engage in a data-rich and interconnected world. Computational thinking refers to the thought processes involved in expressing solutions as computational steps or algorithms that can be carried out by a computer (Cuny, Snyder, & Wing, 2010; Aho, 2011; Lee, 2016).

Fostering an Inclusive Computing Culture

Students will develop skills for building an inclusive and diverse computing culture, which requires strategies for incorporating perspectives from people of different genders, ethnicities, and abilities. Incorporating these perspectives involves understanding the personal, ethical, social, economic, and cultural contexts in which people operate. Considering the needs of diverse users during the design process is essential to producing inclusive computational products.

Collaborating Around Computing

Students will develop skills for collaborating around computing. Collaborative computing is the process of performing a computational task by working in pairs and on teams. Collaborative computing involves asking for the contributions and feedback of others, effective collaboration can lead to better outcomes than working independently. Collaboration requires individuals to navigate and incorporate diverse perspectives, conflicting ideas, disparate skills, and distinct personalities. Students should use collaborative tools to effectively work together and to create complex artifacts.

Recognizing and Defining Computational Problems

Students will develop skills for recognizing and defining computational problems. The ability to recognize appropriate and worthwhile opportunities to apply computation is a skill that develops over time and is central to computing. Solving a problem with a computational approach requires defining the problem, breaking it down into parts, and evaluating each part to determine whether a computational solution is appropriate.

Developing and Using Abstractions

Students will develop skills for developing and using abstractions. Identifying patterns and extracting common features from specific examples to create generalizations form abstractions. Using generalized solutions and parts of solutions designed for broad reuse simplifies the development process by managing complexity.

Creating Computational Artifacts

Students will develop skills for creating computational artifacts. The process of developing computational artifacts embraces both creative expression and the exploration of ideas to create prototypes and solve computational problems. Students create artifacts that are personally relevant or beneficial to their community and beyond. Computational artifacts can be created by

combining and modifying existing artifacts or by developing new artifacts. Examples of computational artifacts include programs, simulations, visualizations, digital animations, robotic systems, and apps.

Testing and Refining Computational Artifacts
Students will develop skills for testing and refining computational artifacts. Testing and refinement is the deliberate and iterative process of improving a computational artifact. This process includes debugging (identifying and fixing errors) and comparing actual outcomes to intended outcomes. Students also respond to the changing needs and expectations of end users and improve the performance, reliability, usability, and accessibility of artifacts.

Communicating About Computing
Students will develop skills for communicating about computing. Communication involves personal expression and exchanging ideas with others. In computer science, students communicate with diverse audiences about the use and effects of computation and the appropriateness of computational choices. Students write clear comments, document their work, and communicate their ideas through multiple forms of media. Clear communication includes using precise language and carefully considering possible audiences.

Equity

The Computer Science standards were developed with the intent of equity, in the form of computer science for all, as a core belief. While equity was a core belief in the development of the Computer Science standards, additional effort at the local and state level with policies and at the school level with curriculum, instruction, and classroom culture is also necessary for equity to play a role in computer science education. The intent of equity in computer science is not to prepare all students to major in computer science at the higher education level and then pursue careers in software development or database management. The intent of equity is to ensure that all students have the basic knowledge that will allow them to productively participate in the world and make well informed decisions about their lives. Equity is not limited to whether classes are available, but also includes how classes are taught, how students are recruited for classes or activities, and how the classroom culture supports diverse learners and promotes the retention of students. The result of equity can be a diverse classroom of students, based on factors such as race, gender, disability, socioeconomic status, and English language proficiency, all of whom have high expectations and feel capable of learning.

Equity has also been considered not only in the composition of the groups writing, advising, and reviewing the Computer Science standards but also in the content of the standards by designing standards that can be engaged in by all students and are flexible enough to allow all students to demonstrate proficiency in multiple ways. The intent of allowing for multiple ways of demonstrating proficiency is further completed by the work of the Virginia Board of Education with the Profile of a Virginia Graduate and the use of locally-developed performance-based assessments.

Organization of the Computer Science Standards of Learning

The Computer Science standards for kindergarten through grade twelve are organized into six strands: Algorithms and Programming, Computing Systems, Cybersecurity, Data and Analysis, Impacts of Computing, and Networks and the Internet. Each grade level is preceded by an overview that describes the major concepts and skills that each student will be expected to understand and demonstrate. The vertically aligned standards are intended to reflect a comprehensive instructional program and document a progression of expected achievement in each of the strands. This organization of standards also reflects the gradual progression in the development of skills.

Algorithms and Programming involves the use of algorithms. An algorithm is a sequence of steps designed to accomplish a specific task. Algorithms are translated into programs, or code, to provide instructions for computing devices. Algorithms and programming control all computing systems, empowering people to communicate with the world in new ways and solve compelling problems. The development process to create meaningful and efficient programs involves choosing which information to use and how to process and store it, breaking apart large problems into smaller ones, recombining existing solutions, and analyzing different solutions.

Computing Systems involves the interaction that people have with a wide variety of computing devices that collect, store, analyze, and act upon information in ways that can affect human capabilities both positively and negatively. The physical components (hardware) and instructions (software) that make up a computing system communicate and process information in digital form. An understanding of hardware and software is useful when troubleshooting a computing system that does not work as intended.

Cybersecurity, also known as information technology security, involves the protection of computers, networks, programs, and data from unauthorized or unintentional access, manipulation, or destruction. Many organizations, such as government, military, corporations, financial institutions, hospitals, and others collect, process, and store significant amounts of data on computing devices. That data is transmitted across multiple networks to other computing devices. The confidential nature of government, financial, and other types of data requires continual monitoring and protection for the sake of continued operation of vital systems and national security.

Data and Analysis involves the data that exist and the computing systems that exist to process that data. The amount of digital data generated in the world is rapidly expanding, so the need to process data effectively is increasingly important. Data is collected and stored so that it can be analyzed to better understand the world and make more accurate predictions.

Impacts of Computing involves the affect that computing has on daily life. Computing affects many aspects of the world in both positive and negative ways at local, national, and global levels. Individuals and communities influence computing through their behaviors and cultural and social interactions, and in turn, computing influences new cultural practices. An informed and responsible person should understand the social implications of the digital world, including equity and access to computing.

Networks and the Internet involves the networks that connect computing systems. Computing devices do not operate in isolation. Networks connect computing devices to share information and resources and are an increasingly integral part of computing. Networks and communication systems provide greater connectivity in the computing world by providing fast, secure communication and facilitating innovation.

Computer Science Foundations

The Computer Science Foundations standards outline the content for a one-year course with an emphasis on computer programming within the context of broader concepts of computer science. The standards build on the concepts of computer science developed in prior grade levels. The standards provide a transition from block-based programming to a text-based programming language and familiarize the student with developing and executing computer programs. Teachers are encouraged to select programming languages and environments, problems, challenges, and activities that are appropriate for their students to successfully meet the objectives of the standards.

Programmable computing tools will be used to facilitate design, analysis, and implementation of computer programs. Students for exploring and creating computer programs, facilitating reasoning and problem solving, and verifying solutions should use these tools.

Computing Systems

CSF.1 The student will

a. compare the structures, functions, and interactions between application software, system software, and hardware; and
b. explore the relationship between hardware and software using the Internet of Things.

Networks and the Internet

CSF.2 The student will model how information is broken down into smaller pieces, transmitted as packets through multiple devices over networks and the Internet, and reassembled at the destination.

CSF.3 The student will explain the role of protocols in transmitting data across networks and the Internet.

CSF.4 The student will evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing.

Cybersecurity

CSF.5 The student will identify and explain ways that sensitive data (assets) can be threatened by malware and other computer attacks, using appropriate terminology.

CSF.6 The student will give examples of ways to protect sensitive data (assets) from malware and other computer attacks and evaluate them according to multiple criteria.

CSF.7 The student will explain typical tradeoffs between usability and security and recommend security measures in a given scenario based on these (or other) tradeoffs.

CSF.8 The student will write or adapt a program to validate its input and to avoid certain kinds of vulnerabilities.

Data and Analysis

CSF.9 The student will evaluate the tradeoffs in how data elements are organized and where data is stored.

CSF.10 The student will create interactive data visualizations using software tools to help others better understand real-world phenomena.

CSF.11 The student will use data analysis tools and techniques to identify patterns in data representing complex systems.

Algorithms and Programming

CSF.12 The student will develop a program working individually and in teams using a text-based language.

CSF.13 The student will identify the expected output of a program given a problem and some input.

CSF.14 The student will design and iteratively develop programs for practical intent or personal expression, incorporating feedback from users.

CSF.15 The student will design and implement algorithms using

a. sequencing of instructions;
b. conditional execution; and
c. iteration.

CSF.16 The student will implement a program that accepts input values, stores them in appropriately named variables, and produces output.

CSF.17 The student will trace the execution of an algorithm, illustrating output and changes in values of named variables.

CSF.18 The student will apply the basic operations used with numeric and non-numeric data types in developing programs.

CSF.19 The student will use predefined functions to simplify the solution of a complex problem.

CSF.20 The student will apply simple algorithms to a collection of data.

CSF.21 The student will create programs

a. demonstrating an understanding that program development is an ongoing process that requires adjusting and debugging along the way; and
b. using version control to create and refine programs.

Impacts of Computing

CSF.22 The student will use tools and methods for collaboration on a project to increase connectivity of people in different cultures and career fields.

CSF.23 The student will evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.

CSF.24 The student will explain the beneficial and harmful effects that intellectual property laws can have on innovation, including the impact of open source software.

CSF.25 The student will explain the privacy concerns related to the collection and generation of data through automated processes that are not always evident to users.