



200 West Baltimore Street • Baltimore, MD 21201 • 410-767-0100 • 410-333-6442 TTY/TDD • marylandpublicschools.org

TO: Members of the State Board of Education

FROM: Karen B. Salmon, Ph.D.

DATE: January 29, 2018

SUBJECT: Computer Science National Standards

PURPOSE:

The purpose of this agenda item is to provide an overview of the Computer Science National Standards and to request the State Board's support in adapting the standards for use in Maryland's local school systems. Attached for your review are the standards, the presentation, and the Governor's Executive Order on computer science.

BACKGROUND/HISTORICAL PERSPECTIVE:

The K-12 Computer Science Framework was released in October of 2016. The framework identifies the following concepts and practices and was used by the Computer Science Teachers Association writing teams to develop the K-12 standards:

Concepts:

- Computing Systems
- Networks and the Internet
- Data and Analysis
- Algorithms and Programming
- Impacts of Computing

Practices:

- Fostering and Inclusive Computing Culture
- Collaborating Around Computing
- Recognizing and Defining Computational Problems
- Developing and Using Abstractions
- Creating Computational Artifacts
- Testing and Refining Computational Artifacts
- Communicating About Computing

The Computer Science Teachers Association (CSTA) released the updated national standards in July of 2017. Educators from Maryland participated in both the writing of the framework as well as the standards. The standards are organized by concept and disaggregated by four age groups: Ages 5-7 (Level 1A); Ages 8-11 (Level 1B); Ages 11-14 (Level 2); and Ages 14-16 (Level 3A). States can either adapt the national standards or use the framework to develop their own computer science standards.

EXECUTIVE SUMMARY:

Maryland State Department of Education staff members are currently working to convene writing teams to review and align the CSTA standards for use in local school systems in Maryland. Drs. Lynne Gilli and Angela Holocker from the Division of Career and College Readiness and the Division of Curriculum, Research, Assessment, and Accountability respectively will convene the first meeting on January 23, 2018. It is anticipated that the updated Maryland standards will be presented to the State Board of Education at the June 2018 Board Meeting for permission to publish.

ACTION: For information and discussion.

KBS/lmg

Attachments



Maryland State Board Meeting

K-12 Computer Science Framework and Standards

January 29, 2018

First Some Terminology

- **Framework**
 - High level, conceptual guide of concepts and practices
 - ex: **K-12 CS Framework**, NRC K-12 Science Framework,
- **Standards**
 - Specific, Measurable, Performance expectations
 - ex: CSTA K-12 CS Standards
- Curriculum framework
 - guides topics and sequence for a curriculum
 - ex: CS Principles framework
- Curriculum
 - what and how to teach a topic
 - ex: Lesson Plans

A general definition of “Framework”: *a basic structure underlying a system or concept*



The K-12 CS Framework Informs:

- Standards,
- Curriculum,
- Professional Development, and
- The Implementation of Computer Science Pathways

Who was Involved in Developing the Framework?

- States and districts

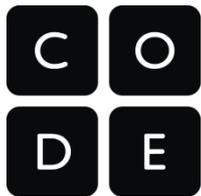
Arkansas	Iowa	New Jersey	<u>Districts:</u>
California	Maryland	North Carolina	Charles County Public Schools
Georgia	Massachusetts	Utah	Chicago Public Schools
Idaho	Nebraska	Washington	NYC Dept of Ed
Indiana	Nevada		San Francisco Unified

- Industry: Google, Amazon, Microsoft, Apple...
- Organizations: College Board, PLTW, Achieve, ISTE, NAF...
- 27 Writers and 25 Advisors from the CS Education Community

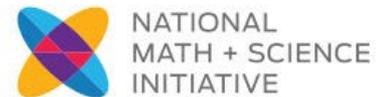


K12 COMPUTER SCIENCE

FRAMEWORK



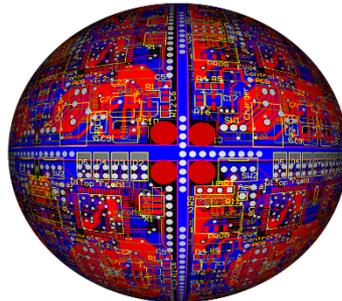
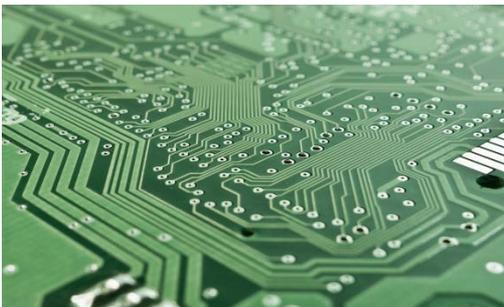
CSTEACHERS.ORG
COMPUTER SCIENCE TEACHERS ASSOCIATION



Computer Science Defined

- The study of computers and algorithmic processes, including their principles, their hardware and software designs, their implementation, and their impact on society.

Association of Computing Machinery (ACM)



Core Concepts and Practices

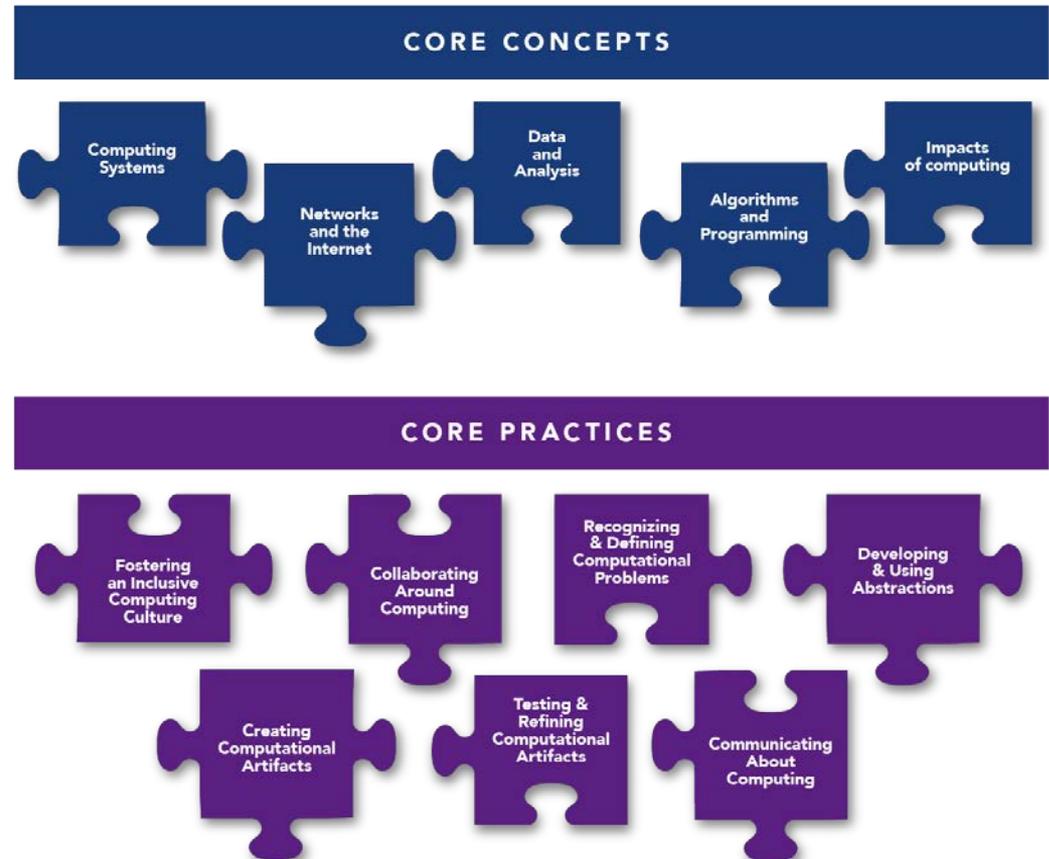
- **5 Concepts**

- Computing Systems
- Networks & the Internet
- Data and Analysis
- Algorithms and Programming
- Impacts of Computing

- **7 Practices**

- Fostering an Inclusive Computing Culture
- Collaborating Around Computing
- Recognizing & Defining Computational Problems
- Developing & Using Abstractions
- Creating Computational Artifacts
- Testing & Refining Computational Artifacts
- Communicating About Computing

THE K-12 COMPUTER SCIENCE FRAMEWORK



Standards: 5 Concepts & 16 Subconcepts

Computing Systems	Devices
	Hardware and Software
	Troubleshooting
Networks and The Internet	Network Communication and Organization
	Cybersecurity
Data and Analysis	Storage
	Collection, Visualization, and Transformation
	Inference and Models
Algorithms and Programming	Algorithms
	Variables
	Control
	Modularity
	Program Development
Impacts of Computing	Culture
	Social Interactions
	Safety, Law, and Ethics

Next Steps for Maryland

- Convene a K-12 Design Team
- Review, Adopt, and/or Adapt the CSTA Standards
 - Include more emphasis on cybersecurity
 - Focus on Maryland's workforce and economic development needs
- Work with Local School Systems on Standards Implementation
- Support the Governors' Partnership for K-12 Computer Science as noted in the Executive Order issued July 14, 2017

Quick Facts About Cybersecurity

- By 2019, the U.S. Department of Labor Statistics projects the number of unfilled cybersecurity jobs to be more than 1.5 million across the nation.
- The need is growing 12 times faster than the overall job market in the U.S.
- Maryland is home to:
 - More than 12,000 IT and cybersecurity companies,
 - 60+ government agencies tasked with protecting our nation from cyber-crime, including the National Security Agency,
 - The National Cybersecurity Center for Excellence,
 - The U.S. Cyber Command, and
 - 17 higher education institutions that have been designated National Academic Centers of Excellence in Cyber Defense.



The State of Maryland

Executive Department

EXECUTIVE ORDER
01.01.2017.27

Computer Science Education and Professional Development

- WHEREAS, On July 14th, 2017 Maryland joined the Governors' Partnership for K-12 Computer Science, a multi-state compact committed to strengthening computer science education for all students, in order to meet the demands of the 21st century workforce and prepare students for the jobs of the future;
- WHEREAS, Creating a talented and diverse pipeline of students with computer expertise is critical to Maryland's economic future;
- WHEREAS, Maryland students must have the skills they need to compete in today's innovation economy;
- WHEREAS, In Maryland, there are more than 20,000 IT related jobs;
- WHEREAS, Maryland's IT industry is a leading force in its economy, and IT and cybersecurity jobs are among our fastest growing positions;
- WHEREAS, Between 2017 and 2027, IT related jobs in Maryland are projected to grow by 12%;
- WHEREAS, The Governor's Workforce Development Board is the Governor of Maryland's chief policy-making body for workforce development;
- WHEREAS, The Governor's Workforce Development Board brings together and focuses various workforce development partners and stakeholders on two key outcomes – a properly prepared workforce that meets the current and future demands of Maryland employers, and providing opportunities for all Marylanders to succeed; and
- WHEREAS, Workforce development should align with the economic and educational goals of the State of Maryland to create a qualified workforce available to employers throughout the State.
- NOW THEREFORE, I, LAWRENCE J. HOGAN, JR., GOVERNOR OF THE STATE OF MARYLAND, BY VIRTUE OF THE AUTHORITY VESTED IN ME BY THE CONSTITUTION AND LAWS OF MARYLAND, HEREBY

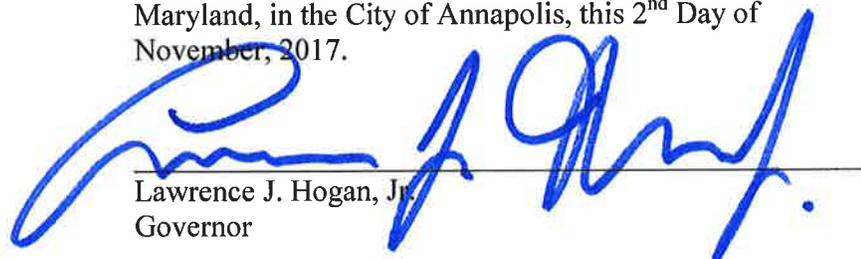
PROCLAIM THE FOLLOWING EXECUTIVE ORDER, EFFECTIVE IMMEDIATELY:

- (1) The Governor's Workforce Development Board Task Force on Cybersecurity and Information Technology shall:
- a. Study opportunities to grow Maryland's economy associated with the computer science and IT industry;
 - b. Focus on developing pathways that meet identified workforce needs in computing fields;
 - c. Determine the skills needed in and challenges for Maryland's talent pipeline;
 - d. Encourage employer partners to invest in Maryland's IT workforce; and
 - e. Create innovative and sustainable ways to address gender and racial disparities in the STEM and IT fields.

(2) Reports and Recommendations.

- a. The Task Force shall issue a report addressing section (1) above.
- b. The report shall be submitted to the Governor, the President of the Senate, and the Speaker of the House of Delegates, no later than June 1, 2018.
- c. The Task Force may issue additional reports as directed by the Governor.

Given Under my Hand and the Great Seal of the State of Maryland, in the City of Annapolis, this 2nd Day of November, 2017.



Lawrence J. Hogan, Jr.
Governor



ATTEST:



John C. Wobensmith
Secretary of State

Progression of Computer Science Teachers Association (CSTA) K-12 Computer Science Standards, Revised 2017

Concept	Subconcept	Level 1A (Ages 5-7) <i>By the end of Grade 2, students will be able to...</i>	Level 1B (Ages 8-11) <i>By the end of Grade 5, students will be able to...</i>	Level 2 (Ages 11-14) <i>By the end of Grade 8, students will be able to...</i>	Level 3A (Ages 14-16) <i>By the end of Grade 10, students will be able to...</i>	Level 3B (Ages 16-18) <i>By the end of Grade 12, students will be able to...</i>
Computing Systems	Devices	1A-CS-01 Select and operate appropriate software to perform a variety of tasks, and recognize that users have different needs and preferences for the technology they use. (P1.1)	1B-CS-01 Describe how internal and external parts of computing devices function to form a system. (P7.2)	2-CS-01 Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices. (P3.3)	3A-CS-01 Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects. (P4.1)	
	Hardware & Software	1A-CS-02 Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware). (P7.2)	1B-CS-02 Model how computer hardware and software work together as a system to accomplish tasks. (P4.4)	2-CS-02 Design projects that combine hardware and software components to collect and exchange data. (P5.1)	3A-CS-02 Compare levels of abstraction and interactions between application software, system software, and hardware layers. (P4.1)	3B-CS-01 Categorize the roles of operating system software. (P7.2)
						3B-CS-02 Illustrate ways computing systems implement logic, input, and output through hardware components. (P7.2)
	Troubleshooting	1A-CS-03 Describe basic hardware and software problems using accurate terminology. (P6.2, P7.2)	1B-CS-03 Determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies. (P6.2)	2-CS-03 Systematically identify and fix problems with computing devices and their components. (P6.2)	3A-CS-03 Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors. (P6.2)	
Networks & The Internet	Network Communication & Organization		1B-NI-04 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. (P4.4)	2-NI-04 Model the role of protocols in transmitting data across networks and the Internet. (P4.4)	3A-NI-04 Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing. (P4.1)	3B-NI-03 Describe the issues that impact network functionality (e.g., bandwidth, load, delay, topology). (P7.2)
	Cybersecurity	1A-NI-04 Explain what passwords are and why we use them, and use strong passwords to protect devices and information from unauthorized access. (P7.3)	1B-NI-05 Discuss real-world cybersecurity problems and how personal information can be protected. (P3.1)	2-NI-05 Explain how physical and digital security measures protect electronic information. (P7.2)	3A-NI-05 Give examples to illustrate how sensitive data can be affected by malware and other attacks. (P7.2)	3B-NI-04 Compare ways software developers protect devices and information from unauthorized access. (P7.2)
Practices		<i>P1. Fostering an Inclusive Computing Culture</i> <i>P2. Collaborating Around Computing</i>	<i>P3. Recognizing and Defining Computational Problems</i> <i>P4. Developing and Using Abstractions</i>	<i>P5. Creating Computational Artifacts</i> <i>P6. Testing and Refining Computational Artifacts</i>	<i>P7. Communicating About Computing</i>	

Progression of Computer Science Teachers Association (CSTA) K-12 Computer Science Standards, Revised 2017

Concept	Subconcept	Level 1A (Ages 5-7)	Level 1B (Ages 8-11)	Level 2 (Ages 11-14)	Level 3A (Ages 14-16)	Level 3B (Ages 16-18)	
Networks & The Internet	Cybersecurity			2-NI-06 Apply multiple methods of encryption to model the secure transmission of information. (P4.4)	3A-NI-06 Recommend security measures to address various scenarios based on factors such as efficiency, feasibility, and ethical impacts. (P3.3)		
					3A-NI-07 Compare various security measures, considering tradeoffs between the usability and security of a computing system. (P6.3)		
					3A-NI-08 Explain tradeoffs when selecting and implementing cybersecurity recommendations. (P7.2)		
Data & Analysis	Storage	1A-DA-05 Store, copy, search, retrieve, modify, and delete information using a computing device and define the information stored as data. (P4.2)	<i>Continuation of standard 1A-DA-05</i>	2-DA-07 Represent data using multiple encoding schemes. (P4.0)	3A-DA-09 Translate between different bit representations of real-world phenomena, such as characters, numbers, and images. (P4.1)		
					3A-DA-10 Evaluate the tradeoffs in how data elements are organized and where data is stored. (P3.3)		
	Collection, Visualization, & Transformation	1A-DA-06 Collect and present the same data in various visual formats. (P7.1, P4.4)	1B-DA-06 Organize and present collected data visually to highlight relationships and support a claim. (P7.1)	2-DA-08 Collect data using computational tools and transform the data to make it more useful and reliable. (P6.3)	3A-DA-11 Create interactive data visualizations using software tools to help others better understand real-world phenomena. (P4.4)	3B-DA-05 Use data analysis tools and techniques to identify patterns in data representing complex systems. (P4.1)	
						3B-DA-06 Select data collection tools and techniques to generate data sets that support a claim or communicate information. (P7.2)	
	Inference & Models	1A-DA-07 Identify and describe patterns in data visualizations, such as charts or graphs, to make predictions. (P4.1)	1B-DA-07 Use data to highlight or propose cause-and-effect relationships, predict outcomes, or communicate an idea. (P7.1)	2-DA-09 Refine computational models based on the data they have generated. (P5.3, P4.4)	3A-DA-12 Create computational models that represent the relationships among different elements of data collected from a phenomenon or process. (P4.4)	3B-DA-07 Evaluate the ability of models and simulations to test and support the refinement of hypotheses. (P4.4)	
	Practices		<i>P1. Fostering an Inclusive Computing Culture</i> <i>P2. Collaborating Around Computing</i>	<i>P3. Recognizing and Defining Computational Problems</i> <i>P4. Developing and Using Abstractions</i>	<i>P5. Creating Computational Artifacts</i> <i>P6. Testing and Refining Computational Artifacts</i>	<i>P7. Communicating About Computing</i>	

Progression of Computer Science Teachers Association (CSTA) K-12 Computer Science Standards, Revised 2017

Concept	Subconcept	Level 1A (Ages 5-7)	Level 1B (Ages 8-11)	Level 2 (Ages 11-14)	Level 3A (Ages 14-16)	Level 3B (Ages 16-18)
Algorithms & Programming	Algorithms	1A-AP-08 Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks. (P4.4)	1B-AP-08 Compare and refine multiple algorithms for the same task and determine which is the most appropriate. (P6.3, P3.3)	2-AP-10 Use flowcharts and/or pseudocode to address complex problems as algorithms. (P4.4, P4.1)	3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests. (P5.2)	
						3B-AP-08 Describe how artificial intelligence drives many software and physical systems. (P7.2)
						3B-AP-09 Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem. (P5.3)
						3B-AP-10 Use and adapt classic algorithms to solve computational problems. (P4.2)
						3B-AP-11 Evaluate algorithms in terms of their efficiency, correctness, and clarity. (P4.2)
	Variables	1A-AP-09 Model the way programs store and manipulate data by using numbers or other symbols to represent information. (P4.4)	1B-AP-09 Create programs that use variables to store and modify data. (P5.2)	2-AP-11 Create clearly named variables that represent different data types and perform operations on their values. (P5.1, P5.2)	3A-AP-14 Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables. (P4.1)	3B-AP-12 Compare and contrast fundamental data structures and their uses. (P4.2)
	Control	1A-AP-10 Develop programs with sequences and simple loops, to express ideas or address a problem. (P5.2)	1B-AP-10 Create programs that include sequences, events, loops, and conditionals. (P5.2)	2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. (P5.1, P5.2)	3A-AP-15 Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made. (P5.2)	3B-AP-13 Illustrate the flow of execution of a recursive algorithm. (P3.2)

Practices	P1. Fostering an Inclusive Computing Culture P2. Collaborating Around Computing	P3. Recognizing and Defining Computational Problems P4. Developing and Using Abstractions	P5. Creating Computational Artifacts P6. Testing and Refining Computational Artifacts	P7. Communicating About Computing
------------------	--	--	--	-----------------------------------

Progression of Computer Science Teachers Association (CSTA) K-12 Computer Science Standards, Revised 2017

Concept	Subconcept	Level 1A (Ages 5-7)	Level 1B (Ages 8-11)	Level 2 (Ages 11-14)	Level 3A (Ages 14-16)	Level 3B (Ages 16-18)
Algorithms & Programming (continued)	Control				3A-AP-16 Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions. (P5.2)	
	Modularity	1A-AP-11 Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions. (P3.2)	1B-AP-11 Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process. (P3.2)	2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. (P3.2)	3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. (P3.2)	3B-AP-14 Construct solutions to problems using student-created components, such as procedures, modules and/or objects. (P5.2)
			1B-AP-12 Modify, remix, or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features. (P5.3)	2-AP-14 Create procedures with parameters to organize code and make it easier to reuse. (P4.1, P4.3)	3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs. (P5.2)	3B-AP-15 Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution. (P4.1)
						3B-AP-16 Demonstrate code reuse by creating programming solutions using libraries and APIs. (P5.3)
	Program Development	1A-AP-12 Develop plans that describe a program's sequence of events, goals, and expected outcomes. (P5.1, P7.2)	1B-AP-13 Use an iterative process to plan the development of a program by including others' perspectives and considering user preferences. (P1.1, P5.1)	2-AP-15 Seek and incorporate feedback from team members and users to refine a solution that meets user needs. (P2.3, P1.1)	3A-AP-19 Systematically design and develop programs for broad audiences by incorporating feedback from users. (P5.1)	3B-AP-17 Plan and develop programs for broad audiences using a software life cycle process. (P5.1)
		1A-AP-13 Give attribution when using the ideas and creations of others while developing programs. (P7.3)	1B-AP-14 Observe intellectual property rights and give appropriate attribution when creating or remixing programs. (P7.3)	2-AP-16 Incorporate existing code, media, and libraries into original programs, and give attribution. (P4.2, P5.2, P7.3)	3A-AP-20 Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries. (P7.3)	3B-AP-18 Explain security issues that might lead to compromised computer programs. (P7.2)
	Practices		<i>P1. Fostering an Inclusive Computing Culture</i> <i>P2. Collaborating Around Computing</i>	<i>P3. Recognizing and Defining Computational Problems</i> <i>P4. Developing and Using Abstractions</i>	<i>P5. Creating Computational Artifacts</i> <i>P6. Testing and Refining Computational Artifacts</i>	<i>P7. Communicating About Computing</i>

Progression of Computer Science Teachers Association (CSTA) K-12 Computer Science Standards, Revised 2017

Concept	Subconcept	Level 1A (Ages 5-7)	Level 1B (Ages 8-11)	Level 2 (Ages 11-14)	Level 3A (Ages 14-16)	Level 3B (Ages 16-18)
Algorithms & Programming (continued)	Program Development	1A-AP-14 Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops. (P6.2)	1B-AP-15 Test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended. (P6.1, P6.2)	2-AP-17 Systematically test and refine programs using a range of test cases. (P6.1)	3A-AP-21 Evaluate and refine computational artifacts to make them more usable and accessible. (P6.3)	3B-AP-19 Develop programs for multiple computing platforms. (P5.2)
			1B-AP-16 Take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation, and review stages of program development. (P2.2)	2-AP-18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. (P2.2)	3A-AP-22 Design and develop computational artifacts working in team roles using collaborative tools. (P2.4)	3B-AP-20 Use version control systems, integrated development environments (IDEs), and collaborative tools and practices (code documentation) in a group software project. (P2.4)
		1A-AP-15 Using correct terminology, describe steps taken and choices made during the iterative process of program development. (P7.2)	1B-AP-17 Describe choices made during program development using code comments, presentations, and demonstrations. (P7.2)	2-AP-19 Document programs in order to make them easier to follow, test, and debug. (P7.2)	3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. (P7.2)	3B-AP-21 Develop and use a series of test cases to verify that a program performs according to its design specifications. (P6.1)
						3B-AP-22 Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality). (P5.3)
						3B-AP-23 Evaluate key qualities of a program through a process such as a code review. (P6.3)
						3B-AP-24 Compare multiple programming languages and discuss how their features make them suitable for solving different types of problems. (P7.2)
Practices		<i>P1. Fostering an Inclusive Computing Culture</i>	<i>P3. Recognizing and Defining Computational Problems</i>	<i>P5. Creating Computational Artifacts</i>	<i>P7. Communicating About Computing</i>	
		<i>P2. Collaborating Around Computing</i>	<i>P4. Developing and Using Abstractions</i>	<i>P6. Testing and Refining Computational Artifacts</i>		

Progression of Computer Science Teachers Association (CSTA) K-12 Computer Science Standards, Revised 2017

Concept	Subconcept	Level 1A (Ages 5-7)	Level 1B (Ages 8-11)	Level 2 (Ages 11-14)	Level 3A (Ages 14-16)	Level 3B (Ages 16-18)	
Impacts of Computing	Culture	1A-IC-16 Compare how people live and work before and after the implementation or adoption of new computing technology. (P7.0)	1B-IC-18 Discuss computing technologies that have changed the world, and express how those technologies influence, and are influenced by, cultural practices. (P7.1)	2-IC-20 Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. (P7.2)	3A-IC-24 Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices. (P1.2)	3B-IC-25 Evaluate computational artifacts to maximize their beneficial effects and minimize harmful effects on society. (P6.1, 1.2)	
			1B-IC-19 Brainstorm ways to improve the accessibility and usability of technology products for the diverse needs and wants of users. (P1.2)	2-IC-21 Discuss issues of bias and accessibility in the design of existing technologies. (P1.2)	3A-IC-25 Test and refine computational artifacts to reduce bias and equity deficits. (P1.2)	3B-IC-26 Evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society. (P1.2)	
					3A-IC-26 Demonstrate ways a given algorithm applies to problems across disciplines. (P3.1)	3B-IC-27 Predict how computational innovations that have revolutionized aspects of our culture might evolve. (P7.2)	
	Social Interactions	1A-IC-17 Work respectfully and responsibly with others online. (P2.1)	1B-IC-20 Seek diverse perspectives for the purpose of improving computational artifacts. (P1.1)	2-IC-22 Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact. (P2.4, P5.2)	3A-IC-27 Use tools and methods for collaboration on a project to increase connectivity of people in different cultures and career fields. (P2.4)		
	Safety, Law, & Ethics			1B-IC-21 Use public domain or creative commons media, and refrain from copying or using material created by others without permission. (P7.3)		3A-IC-28 Explain the beneficial and harmful effects that intellectual property laws can have on innovation. (P7.3)	3B-IC-28 Debate laws and regulations that impact the development and use of software. (P3.3, 7.3)
			1A-IC-18 Keep login information private, and log off of devices appropriately. (P7.3)		2-IC-23 Describe tradeoffs between allowing information to be public and keeping information private and secure. (P7.2)	3A-IC-29 Explain the privacy concerns related to the collection and generation of data through automated processes that may not be evident to users. (P7.2)	
					3A-IC-30 Evaluate the social and economic implications of privacy in the context of safety, law, or ethics. (P7.3)		

P1. Practices	P2. Fostering an Inclusive Computing Culture	P3. Collaborating Around Computing	P4. Recognizing and Defining Computational Problems	P5. Developing and Using Abstractions	P6. Creating Computational Artifacts	P7. Testing and Refining Computational Artifacts	P8. Communicating About Computing
----------------------	---	---	--	--	---	---	--

