Computer Science
Teachers Association

K-12 Computer Science Standards

Revised 2017

The CSTA Standards Task Force

Deborah Seehorn, Co-Chair
North Carolina Department of Public Instruction (Retired)

Tammy Pirmann, Co-Chair
School District of Springfield Township

Todd Lash, Elementary Grades Team Lead
University of Illinois

Bryan Twarek, Middle Grades Team Lead
San Francisco Unified School District

Daniel Moix, High School Team Lead
Arkansas School for Mathematics, Sciences & Arts

Leticia Batista
Oxnard School District

Julia Bell
Walters State Community College

Chris Kuszmaul
Palo Alto High School

Dianne O'Grady-Cunniff
Charles County Public Schools

Minsoo Park
Countryside School

Lori Pollock
University of Delaware

Meg Ray
Cornell Tech

Dylan Ryder
The School at Columbia University

Vicky Sedgwick
St. Martin's Episcopal School

Grant Smith
Launch CS

Chinma Uche
Greater Hartford Academy of Mathematics and Science

The K–12 Computer Science Framework, led by the Association for Computing Machinery, Code.org, Computer Science Teachers Association, Cyber Innovation Center, and National Math and Science Initiative in partnership with states and districts, informed the development of this work.

About the CSTA K-12 Computer Science Standards

Computer science and the technologies it enables rest at the heart of our economy and the way we live our lives. To be well-educatec citizens in a computing-intensive world and to be prepared for careers in the 21st century, our students must have a clear understanding of the principles and practices of computer science. The CSTA K–12 Computer Science Standards delineate a core set of learning objectives designed to provide the foundation for a complete computer science curriculum and its implementation at the K–12 level. To this end, the CSTA Standards:

- Introduce the fundamental concepts of computer science to all students, beginning at the elementary school level.
- Present computer science at the secondary school level in a way that can fulfill a computer science, math, or science graduation credit.
- Encourage schools to offer additional secondary-level computer science courses that will allow interested students to study facets of computer science in more depth and prepare them for entry into the work force or college.
- Increase the availability of rigorous computer science for all students, especially those who are members of underrepresented groups.

The standards have been written by educators to be coherent and comprehensible to teachers, administrators, and policy makers.

Levels 1A, 1B, 2, and 3A are the computer science standards for **ALL students**. The Level 3B standards are intended for students who wish to pursue the study of computer science in high school beyond what is required for all students (specialty or elective courses).

Connection to the *K-12 Computer Science Framework*

The K–12 Computer Science Framework (k12cs.org) provides overarching, high-level guidance per grade bands, while the standards provide detailed, measurable student performance expectations. The Framework was considered as a primary input for the standards development process.

The CSTA Standards Revision Task Force crafted standards by combining concept statements and practices from the Framework. It also used descriptive material from the Framework when writing examples and clarifying statements to accompany the standards.

Concepts

1. Computing Systems
2. Networks and the Internet
3. Data and Analysis
4. Algorithms and Programming
5. Impacts of Computing

Practices

1. Fostering an Inclusive Computing Culture
2. Collaborating Around Computing
3. Recognizing and Defining Computational Problems
4. Developing and Using Abstractions
5. Creating Computational Artifacts
6. Testing and Refining Computational Artifacts
7. Communicating About Computing

Level 1A: Grades K-2 (Ages 5-7)

Computing Systems

Identifier	Standard	Subconcept	Practice
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.	Devices	1.1
1A-CS-02	Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware).	Hardware & Software	7.2
1A-CS-03	Describe basic hardware and software problems using accurate terminology.	Troubleshooting	6.2, 7.2

Networks and the Internet

1A-NI-04	Explain what passwords are and why we use them, and use strong passwords to protect devices and information from unauthorized access.	Cybersecurity	7.3

Data and Analysis

1A-DA-05	Store, copy, search, retrieve, modify, and delete information using a computing device and define the information stored as data.	Storage	4.2
1A-DA-06	Collect and present the same data in various visual formats.	Collection Visualization & Transformation	7.1, 4.4
1A-DA-07	Identify and describe patterns in data visualizations, such as charts or graphs, to make predictions.	Inference & Models	4.1

Algorithms and Programming

1A-AP-08	Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.	Algorithms	4.4
1A-AP-09	Model the way programs store and manipulate data by using numbers or other symbols to represent information.	Variables	4.4

1A-AP-10	Develop programs with sequences and simple loops, to express ideas or address a problem.	Control	5.2
1A-AP-11	Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.	Modularity	3.2
1A-AP-12	Develop plans that describe a program's sequence of events, goals, and expected outcomes.	Program Development	5.1, 7.2
1A-AP-13	Give attribution when using the ideas and creations of others while developing programs.	Program Development	7.3
1A-AP-14	Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.	Program Development	6.2
1A-AP-15	Using correct terminology, describe steps taken and choices made during the iterative process of program development.	Program Development	7.2

Impacts of Computing

1A-IC-16	Compare how people live and work before and after the implementation or adoption of new computing technology.	Culture	7
1A-IC-17	Work respectfully and responsibly with others online.	Social Interactions	2.1
1A-IC-18	Keep login information private, and log off of devices appropriately.	Safety Law & Ethics	7.3

Level 1B: Grades 3-5 (Ages 8-11)

Computing Systems

Identifier	Standard	Subconcept	Practice
1B-CS-01	Describe how internal and external parts of computing devices function to form a system.	Devices	7.2
1B-CS-02	Model how computer hardware and software work together as a system to accomplish tasks.	Hardware & Software	4.4
1B-CS-03	Determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies.	Troubleshooting	6.2

Networks and the Internet

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.	Network Communication & Organization	4.4
1B-NI-05	Discuss real-world cybersecurity problems and how personal information can be protected.	Cybersecurity	3.1

Data and Analysis

1B-DA-06	Organize and present collected data visually to highlight relationships and support a claim.	Collection Visualization & Transformation	7.1
1B-DA-07	Use data to highlight or propose cause-and-effect relationships, predict outcomes, or communicate an idea.	Inference & Models	7.1

Algorithms and Programming

1B-AP-08	Compare and refine multiple algorithms for the same task and determine which is the most appropriate.	Algorithms	6.3, 3.3
1B-AP-09	Create programs that use variables to store and modify data.	Variables	5.2

1B-AP-10	Create programs that include sequences, events, loops, and conditionals.	Control	5.2
1B-AP-11	Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.	Modularity	3.2
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.	Modularity	5.3
1B-AP-13	Use an iterative process to plan the development of a program by including others' perspectives and considering user preferences.	Program Development	1.1, 5.1
1B-AP-14	Observe intellectual property rights and give appropriate attribution when creating or remixing programs.	Program Development	5.2, 7.3
1B-AP-15	Test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended.	Program Development	6.1, 6.2
1B-AP-16	Take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation, and review stages of program development.	Program Development	2.2
1B-AP-17	Describe choices made during program development using code comments, presentations, and demonstrations.	Program Development	7.2

Impacts of Computing

1B-IC-18	Discuss computing technologies that have changed the world, and express how those technologies influence, and are influenced by, cultural practices.	Culture	3.1
1B-IC-19	Brainstorm ways to improve the accessibility and usability of technology products for the diverse needs and wants of users.	Culture	1.2
1B-IC-20	Seek diverse perspectives for the purpose of improving computational artifacts.	Social Interactions	1.1
1B-IC-21	Use public domain or creative commons media, and refrain from copying or using material created by others without permission.	Safety Law & Ethics	7.3

Level 2: Grades 6-8 (Ages 11-14)

Computing Systems

Identifier	Standard	Subconcept	Practice
2-CS-01	Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices.	Devices	3.3
2-CS-02	Design projects that combine hardware and software components to collect and exchange data.	Hardware & Software	5.1
2-CS-03	Systematically identify and fix problems with computing devices and their components.	Troubleshooting	6.2

Networks and the Internet

2-NI-04	Model the role of protocols in transmitting data across networks and the Internet.	Network Communication & Organization	4.4
2-NI-05	Explain how physical and digital security measures protect electronic information.	Cybersecurity	7.2
2-NI-06	Apply multiple methods of encryption to model the secure transmission of information.	Cybersecurity	4.4

Data and Analysis

2-DA-07	Represent data using multiple encoding schemes.	Storage	4
2-DA-08	Collect data using computational tools and transform the data to make it more useful and reliable.	Collection Visualization & Transformation	6.3
2-DA-09	Refine computational models based on the data they have generated.	Inference & Models	5.3, 4.4

Algorithms and Programming

2-AP-10	Use flowcharts and/or pseudocode to address complex problems as algorithms.	Algorithms	4.4, 4.1

2-AP-11	Create clearly named variables that represent different data types and perform operations on their values.	Variables	5.1, 5.2
2-AP-12	Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.	Control	5.1, 5.2
2-AP-13	Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.	Modularity	3.2
2-AP-14	Create procedures with parameters to organize code and make it easier to reuse.	Modularity	4.1, 4.3
2-AP-15	Seek and incorporate feedback from team members and users to refine a solution that meets user needs.	Program Development	2.3, 1.1
2-AP-16	Incorporate existing code, media, and libraries into original programs, and give attribution.	Program Development	4.2, 5.2, 7.3
2-AP-17	Systematically test and refine programs using a range of test cases.	Program Development	6.1
2-AP-18	Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.	Program Development	2.2
2-AP-19	Document programs in order to make them easier to follow, test, and debug.	Program Development	7.2

Impacts of Computing

2-IC-20	Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.	Culture	7.2
2-IC-21	Discuss issues of bias and accessibility in the design of existing technologies.	Culture	1.2
2-IC-22	Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact.	Social Interactions	2.4, 5.2
2-IC-23	Describe tradeoffs between allowing information to be public and keeping information private and secure.	Safety Law & Ethics	7.2

Level 3A: Grades 9-10 (Ages 14-16)

Computing Systems

Identifier	Standard	Subconcept	Practice
3A-CS-01	Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.	Devices	4.1
3A-CS-02	Compare levels of abstraction and interactions between application software, system software, and hardware layers.	Hardware & Software	4.1
3A-CS-03	Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.	Troubleshooting	6.2

Networks and the Internet

3A-NI-04	Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing.	Network Communication & Organization	4.1
3A-NI-05	Give examples to illustrate how sensitive data can be affected by malware and other attacks.	Network Communication & Organization	7.2
3A-NI-06	Recommend security measures to address various scenarios based on factors such as efficiency, feasibility, and ethical impacts.	Cybersecurity	3.3
3A-NI-07	Compare various security measures, considering tradeoffs between the usability and security of a computing system.	Network Communication & Organization	6.3
3A-NI-08	Explain tradeoffs when selecting and implementing cybersecurity recommendations.	Cybersecurity	7.2

Data and Analysis

3A-DA-09	Translate between different bit representations of real-world phenomena, such as characters, numbers, and images.	Storage	4.1

3A-DA-10	Evaluate the tradeoffs in how data elements are organized and where data is stored.	Storage	3.3
3A-DA-11	Create interactive data visualizations using software tools to help others better understand real-world phenomena.	Collection Visualization & Transformation	4.4
3A-DA-12	Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.	Inference & Models	4.4

Algorithms and Programming

3A-AP-13	Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.	Algorithms	5.2
3A-AP-14	Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.	Variables	4.1
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.	Control	5.2
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.	Control	5.2
3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.	Control	3.2
3A-AP-18	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.	Modularity	5.2
3A-AP-19	Systematically design and develop programs for broad audiences by incorporating feedback from users.	Modularity	5.1
3A-AP-20	Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries.	Program Development	7.3
3A-AP-21	Evaluate and refine computational artifacts to make them more usable and accessible.	Program Development	6.3

3A-AP-22	Design and develop computational artifacts working in team roles using collaborative tools.	Program Development	2.4
3A-AP-23	Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.	Program Development	7.2

Impacts of Computing

3A-IC-24	Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.	Culture	1.2
3A-IC-25	Test and refine computational artifacts to reduce bias and equity deficits.	Culture	1.2
3A-IC-26	Demonstrate ways a given algorithm applies to problems across disciplines.	Culture	3.1
3A-IC-27	Use tools and methods for collaboration on a project to increase connectivity of people in different cultures and career fields.	Social Interactions	2.4
3A-IC-28	Explain the beneficial and harmful effects that intellectual property laws can have on innovation.	Safety Law & Ethics	7.3
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.	Safety Law & Ethics	7.2
3A-IC-30	Evaluate the social and economic implications of privacy in the context of safety, law, or ethics.	Safety Law & Ethics	7.3

Level 3B: Grades 11-12 (Ages 16-18)

Computing Systems

Identifier	Standard	Subconcept	Practice
3B-CS-01	Categorize the roles of operating system software.	Hardware & Software	7.2
3B-CS-02	Illustrate ways computing systems implement logic, input, and output through hardware components.	Troubleshooting	7.2

Networks and the Internet

3B-NI-03	Describe the issues that impact network functionality (e.g., bandwidth, load, delay, topology).	Network Communication & Organization	7.2
3B-NI-04	Compare ways software developers protect devices and information from unauthorized access.	Cybersecurity	7.2

Data and Analysis

3B-DA-05	Use data analysis tools and techniques to identify patterns in data representing complex systems.	Collection Visualization & Transformation	4.1
3B-DA-06	Select data collection tools and techniques to generate data sets that support a claim or communicate information.	Collection Visualization & Transformation	7.2
3B-DA-07	Evaluate the ability of models and simulations to test and support the refinement of hypotheses.	Inference & Models	4.4

Algorithms and Programming

3B-AP-08	Describe how artificial intelligence drives many software and physical systems.	Algorithms	7.2
3B-AP-09	Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.	Algorithms	5.3
3B-AP-10	Use and adapt classic algorithms to solve computational problems.	Algorithms	4.2
3B-AP-11	Evaluate algorithms in terms of their efficiency, correctness, and clarity.	Algorithms	4.2
3B-AP-12	Compare and contrast fundamental data structures and their uses.	Variables	4.2
3B-AP-13	Illustrate the flow of execution of a recursive algorithm.	Control	3.2
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.	Modularity	5.2
3B-AP-15	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.	Modularity	4.1
3B-AP-16	Demonstrate code reuse by creating programming solutions using libraries and APIs.	Modularity	5.3
3B-AP-17	Plan and develop programs for broad audiences using a software life cycle process.	Program Development	5.1
3B-AP-18	Explain security issues that might lead to compromised computer programs.	Program Development	7.2
3B-AP-19	Develop programs for multiple computing platforms.	Program Development	5.2
3B-AP-20	Use version control systems, integrated development environments (IDEs), and collaborative tools and practices (code documentation) in a group software project.	Program Development	2.4
3B-AP-21	Develop and use a series of test cases to verify that a program performs according to its design specifications.	Program Development	6.1
3B-AP-22	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).	Program Development	5.3

3B-AP-23	Evaluate key qualities of a program through a process such as a code review.	Program Development	6.3
3B-AP-24	Compare multiple programming languages and discuss how their features make them suitable for solving different types of problems.	Program Development	7.2

Impacts of Computing

3B-IC-25	Evaluate computational artifacts to maximize their beneficial effects and minimize harmful effects on society.	Culture	6.1, 1.2
3B-IC-26	Evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society.	Culture	1.2
3B-IC-27	Predict how computational innovations that have revolutionized aspects of our culture might evolve.	Culture	7.2
3B-IC-28	Debate laws and regulations that impact the development and use of software.	Safety Law & Ethics	3.3, 7.3

Association for Computing Machinery

The CSTA K-12 Computer Science Standards are created and maintained by educator members of the Computer Science Teachers Association (CSTA). The Association for Computing Machinery (ACM) founded CSTA as part of its commitment to K-12 computer science education.