washington
STATE LEARNING STANDARDS

kindergarten – 12th grade

Computer Science

K-2 Standards

SUPERINTENDENT OF PUBLIC INSTRUCTION WASHINGTON

Revised 2018

License

Attribution

The CSTA K–12 Computer Science Standards are created and maintained by 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. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Suggested citation: Computer Science Teachers Association (2017). CSTA K–12 Computer Science Standards, Revised 2017. Retrieved from http://www.csteachers.org/standards.

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.

The CSTA Standards Revision Task Force crafted standards by combining concept statements and practices from the Framework. The Task Force also used descriptive material from the Framework when writing examples and clarifying statements to accompany the standards. The glossary referenced in the navigation header links directly to the Framework's glossary.

For more information about the Framework, please visit k12cs.org.

Legend for Identifiers

Unique Numbering System for the Washington Computer Science K–12 Learning Standards

To help organize and track each individual standard, a unique identifier was developed. An example appears below:

Level	Framework Concept	Number	Computer Science K–12 Learning Standard
Grades 6-8	Algorithms and Programming	17	Systematically test and refine programs using a range of test cases.
2	**AP**	**17**	**Identifier: 2-AP-17**

Use the following legend to interpret the unique identifier for each Computer Science K–12 Learning Standard:

The identifier code corresponds to: **Level – Concept – Number**		
Identifier Code		**Key**
Levels	1A	Grades K–2
	1B	Grades 3–5
	2	Grades 6–8
	3A	Grades 9–10
	3B	Grades 11–12
Concepts	CS	Computing Systems
	NI	Networks and the Internet
	DA	Data and Analysis
	AP	Algorithms and Programming
	IC	Impacts of Computing

Integrated into classroom activities through practices:

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

Figure 1: Standards Identifier Code –
Computer Science Teachers Association K–12 Computer Science Standards (2017)
Retrieved from http://www.csteachers.org

K-2 Standards

Identifier	Level 1A: K–2
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. (P 1.1)
1A-CS-02	Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware). (P 7.2)
1A-CS-03	Describe basic hardware and software problems using accurate terminology. (P 6.2, P 7.2)
1A-NI-04	Explain what passwords are and why we use them, and use strong passwords to protect devices and information from unauthorized access. (P 7.3)
1A-DA-05	Store, copy, search, retrieve, modify, and delete information using a computing device and define the information stored as data. (P 4.2)
1A-DA-06	Collect and present the same data in various visual formats. (P 7.1, P 4.4)
1A-DA-07	Identify and describe patterns in data visualizations, such as charts or graphs, to make predictions. (P 4.1)
1A-AP-08	Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks. (P. 4.4)
1A-AP-09	Model the way programs store and manipulate data by using numbers or other symbols to represent information. (P 4.4)
1A-AP-10	Develop programs with sequences and simple loops, to express ideas or address a problem. (P. 5.2)
1A-AP-11	Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions. (P. 3.2)
1A-AP-12	Develop plans that describe a program's sequence of events, goals, and expected outcomes. (P. 5.1, P. 7.2)
1A-AP-13	Give attribution when using the ideas and creations of others while developing programs. (P. 7.3)
1A-AP-14	Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops. (P. 6.2)
1A-AP-15	Using correct terminology, describe steps taken and choices made during the iterative process of program development. (P. 7.2)
1A-IC-16	Compare how people live and work before and after the implementation or adoption of new computing technology. (P. 7)
1A-IC-17	Work respectfully and responsibly with others online. (P. 2.1)
1A-IC-18	Keep login information private, and log off of devices appropriately. (P. 7.3)

Computer Science Glossary

The following glossary includes definitions of terms used in the statements in the Washington Computer Science K–12 Learning Standards. These terms are intended to increase teacher understanding and decrease biased language.

abstraction *(process):* The process of reducing complexity by focusing on the main idea. By hiding details irrelevant to the question at hand and bringing together related and useful details, abstraction reduces complexity and allows one to focus on the problem. In elementary classrooms, abstraction is hiding unnecessary details to make it easier to think about a problem.

abstraction *(product)*: A new representation of a thing, a system, or a problem that helpfully reframes a problem by hiding details irrelevant to the question at hand. [MA-DLCS]

abstraction (Code.org K–5) Pulling out specific differences to make one solution work for multiple problems.

algorithm: A step-by-step process to complete a task.

A list of steps to finish a task. A set of instructions that can be performed with or without a computer. For example, the collection of steps to make a peanut butter and jelly sandwich is an algorithm. (Code.org K–5)

artifact: Anything created by a human. *See "computational artifact" for the computer science-specific*

Block-based programming language: (Code.org K–5) Any programming language that lets users create programs by manipulating "blocks" or graphical programing elements, rather than writing code using text. Examples include Code Studio, Scratch, and Swift. (Sometimes called visual coding, drag and drop programming, or graphical programming blocks)

bug: An error in a software program. It may cause a program to unexpectedly quit or behave in an unintended manner. [TechTerms] The process of removing errors (bugs) is called debugging.

An error in a program that prevents the program from running as expected. (Code.org K–5)

code: Any set of instructions expressed in a programming language. [MA-DLCS] One or more commands or algorithm(s) designed to be carried out by a computer. (Code.org K–5) See also: program

computational artifact: Anything created by a human using a computational thinking process and a computing device. A computational artifact can be, but is not limited to, a program, image, audio, video, presentation, or web page file.

computational thinking: Mental processes and strategies that include: decomposition, pattern matching, abstraction, algorithms (decomposing problems into smaller, more manageable problems, finding repeating patterns, abstracting specific differences to make one solution work for multiple problems, and creating step-by-step algorithms). (Code.org K–5)

computer science: Using the power of computers to solve problems. (Code.org K–5)

data: Information. Often, quantities, characters, or symbols that are the inputs and outputs of computer programs. (Code.org K–5)

debugging: Finding and fixing errors in programs. (Code.org K–5)

decompose: Break a problem down into smaller pieces. (Code.org K–5)

event: An action that causes something to happen. (Code.org K–5)

function: A type of procedure or routine. Some programming languages make a distinction between a function, which returns a value, and a procedure, which performs some operation, but does not return a value. [MA-DLCS] *Note: This definition differs from that used in math.* A piece of code that you can easily call over and over again. Functions are sometimes called 'procedures.' (Code.org K–5)

hardware: The physical components that make up a computing system, computer, or computing device. [MA-DLCS]

Internet: The global collection of computer networks and their connections, all using shared protocols to communicate [CAS-Prim] A group of computers and servers that are connected to each other. (Code.org K–5)

iterative: Involving the repeating of a process with the aim of approaching a desired goal, target, or result. [MA-DLCS]

loop: A programming structure that repeats a sequence of instructions as long as a specific condition is true. [TechTerms]

looping: Repetition, using a loop. The action of doing something over and over again. (Code.org K–5)

model: A representation of (some part of) a problem or a system. (Modeling (v): the act of creating a model) [MA-DLCS] *Note: This definition differs from that used in science.*

network: A group of computing devices (personal computers, phones, servers, switches, routers, and so on) connected by cables or wireless media for the exchange of information and resources.

operation: An action, resulting from a single instruction, that changes the state of data. [Dictionary.com]

pattern matching: Finding similarities between things. (Code.org K–5)

persistence: Trying again and again, even when something is very hard. (Code.org K–5)

procedure: An independent code module that fulfills some concrete task and is referenced within a larger body of source code. This kind of code item can also be called a function or a subroutine. The fundamental role of a procedure is to offer a single point of reference for some small goal or task that the developer or programmer can trigger by invoking the procedure itself. A procedure may also be referred to as a function, subroutine, routine, method or subprogram. [Techopedia]

program; programming *(n)*: A set of instructions that the computer executes in order to achieve a particular objective. [MA-DLCS] **program** *(v)*: To produce a program by programming. An algorithm that has been coded into something that can be run by a machine. (Code.org K–5)

programming: The craft of analyzing problems and designing, writing, testing, and maintaining programs to solve them. [MA-DLCS] The art of creating a program. (Code.org K–5)

protocol: The special set of rules that end points in a telecommunication connection use when they communicate. Protocols specify interactions between the communicating entities. [TechTarget]

software: Programs that run on a computer system, computer, or other computing device.

structure: A general term used in the framework to discuss the concept of encapsulation without specifying a particular paradigm.

switch: A high-speed device that receives incoming data packets and redirects them to their destination on a local area network (LAN). [Techopedia]

system: A collection of elements or components that work together for a common purpose. [TechTarget] A collection of computing hardware and software integrated for the purpose of accomplishing shared tasks.

troubleshooting: A systematic approach to problem solving that is often used to find and resolve a problem, error, or fault within software or a computer system. [Techopedia, TechTarget]

user: A person for whom a hardware or software product is designed (as distinguished from the developers). [TechTarget]

Key to sources of multiple definitions in this glossary:

CAS-Prim: Computing at School. Computing in the national curriculum: A guide for primary teachers (http://www.computingatschool.org.uk/data/uploads/CASPrimaryComputing.pdf)

Code.org: Creative Commons License (CC BY-NC-SA 4.0) (https://code.org/curriculum/docs/k-5/glossary)

Computer Science Teachers Association: CSTA K–12 Computer Science Standards (2011) https://csta.acm.org/Curriculum/sub/K12Standards.html

MA-DLCS: Massachusetts Digital Literacy and Computer Science Standards, Glossary (Draft, December 2015)

Techopedia: Techopedia. (https://www.techopedia.com/dictionary)

TechTarget: TechTarget Network. (http://www.techtarget.com/network)

TechTerms: Tech Terms Computer Dictionary. (http://www.techterms.com)

Chris Reykdal • State Superintendent
Office of Superintendent of Public Instruction
Old Capitol Building • P.O. Box 47200
Olympia, WA 98504-7200