Legend for Identifiers

Unique Numbering System for the 2016 K–12 Computer Science Learning Standards
To help organize and track each individual standard, a unique identifier was developed. An example appears below:

Grades	Identifier	Computer Science K–12 Learning Standard	Framework Concept	Framework Practice
9–10	**3A-A-2-1**	Design and develop a software artifact working in a team.	Algorithms and Programming	Collaborating

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

The identifier code corresponds to: **Level – Concept – Practice – Identifier**		
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	A	Algorithms and Programming
	C	Computing Systems
	D	Data and Analysis
	I	Impacts of Computing
	N	Networks and the Internet
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 4: Standards Identifier Code - Interim Computer Science Teachers Association K–12 Computer Science Standards (2016) Retrieved from http://www.csteachers.org

11–12	Level 3B
3B-A-2-1	Use version control systems, integrated development environments (IDEs), and collaborating tools and practices (code documentation) in a group software project.
3B-A-2-2	Demonstrate software life cycle processes (e.g., spiral, waterfall) by participating on software project teams (e.g., community service project with real-world clients).
3B-A-7-3	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
3B-A-7-4	Explain security issues that might lead to compromised computer programs (e.g., circular references, ambiguous program calls, lack of error checking and field size checking).
3B-A-7-5	Compare a variety of programming languages and identify features that make them useful for solving different types of problems and developing different kinds of systems (e.g., declarative, logic, parallel, functional, compiled, interpreted, real-time).
3B-A-7-6	Describe how artificial intelligence drives many software and physical systems (e.g., autonomous robots, computer vision, pattern recognition, text analysis).
3B-A-5-7	Decompose a problem by creating new data types, functions, or classes.
3B-A-5-8	Demonstrate code reuse by creating programming solutions using libraries and APIs (e.g., graphics libraries, maps API).
3B-A-5-9	Implement an AI algorithm to play a game against a human opponent or solve a problem.
3B-A-5-10	Develop programs for multiple computing platforms (e.g., computer desktop, web, mobile).
3B-A-4-11	Critically analyze classic algorithms (e.g., sorting, searching) and use in different contexts, adapting as appropriate.
3B-A-4-12	Evaluate algorithms (e.g., sorting, searching) in terms of their efficiency, correctness, and clarity.
3B-A-4-13	Compare and contrast fundamental data structures and their uses (e.g., lists, maps, arrays, stacks, queues, trees, graphs).
3B-A-4-14	Discuss issues that arise when breaking large-scale problems down into parts that must be processed simultaneously on separate systems (e.g., cloud computing, parallelization, concurrency).
3B-A-3-15	Provide examples of computationally solvable problems and difficult-to-solve problems.
3B-A-3-16	Explain the value of heuristic algorithms (discovery methods) to approximating solutions for difficult-to-solve computational problems.

11–12	Level 3B
3B-A-3-17	Decompose a large-scale computational problem by identifying generalizable patterns and applying them in a solution.
3B-A-3-18	Illustrate the flow of execution of a recursive algorithm.
3B-A-3-19	Describe how parallel processing can be used to solve large problems (e.g., SETI at Home, FoldIt).
3B-A-3-20	Develop and use a series of test cases to verify that a program performs according to its design specifications.
3B-A-6-21	Evaluate key qualities of a program (e.g., correctness, usability, readability, efficiency, portability, scalability) through a process such as a code review.
3B-C-7-22	Explain the role of operating systems (e.g., how programs are stored in memory, how data is organized/retrieved, how processes are managed and multi-tasked).
3B-C-7-23	Identify the functionality of various categories of hardware components and communication between them (e.g., physical layers, logic gates, chips, input and output devices).
3B-D-4-24	Use data analysis to identify significant patterns in complex systems (e.g., take existing data sets and make sense of them).
3B-D-4-25	Discuss how data sequences (e.g., binary, hexadecimal, octal) can be interpreted in a variety of forms (e.g., instructions, numbers, text, sound, image).
3B-D-4-26	Evaluate the ability of models and simulations to formulate, refine, and test hypotheses.
3B-D-4-27	Identify mathematical and computational patterns through modeling and simulation (e.g., regression, Runge-Kutta, queueing theory, discrete event simulation).
3B-D-1-28	Use various data collection techniques for different types of problems (e.g., mobile device, GPS, user surveys, embedded system sensors, open data sets, social media data sets).
3B-D-3-29	Explore security policies by implementing and comparing encryption and authentication strategies (e.g., secure coding, safeguarding keys).
3B-I-7-30	Develop criteria to evaluate the beneficial and harmful effects of computing innovations on people and society.
3B-I-5-31	Select, observe, and contribute to global Collaborating in the development of a computational artifact (e.g., contribute the resolution of a bug in an open-source project hosted on GitHub).

11–12	Level 3B
3B-I-1-32	Design and implement a study that evaluates or predicts how computation has revolutionized an aspect of our culture and how it might evolve (e.g., education, healthcare, art/entertainment, energy).
3B-I-1-33	Debate laws and regulations that impact the development and use of software.
3B-I-1-34	Evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society.
3B-N-4-35	Simulate and discuss the issues (e.g., bandwidth, load, delay, topology) that impact network functionality (e.g., use free network simulators).