

## 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 1: Standards Identifier Code - Interim Computer Science Teachers Association K–12 Computer Science Standards (2016) Retrieved from http://www.csteachers.org

| K–2 | **Level 1A** |
| --- | --- |
| 1A-A-7-1 | Give credit when using code, music, or pictures (for example) that were created by others. |
| 1A-A-5-2 | Construct programs, to accomplish a task or as a means of creative expression, which include sequencing, events, and simple loops, using a block-based visual programming language, both independently and collaboratively (e.g., pair programming). |
| 1A-A-5-3 | Plan and create a design document to illustrate thoughts, ideas, and stories in a sequential (step-by-step) manner (e.g., story map, storyboard, sequential graphic organizer). |
| 1A-A-4-4 | Use numbers or other symbols to represent data (e.g., thumbs up/down for yes/no, color by number, arrows for direction, encoding/decoding a word using numbers or pictographs). |
| 1A-A-3-5 | Decompose (break down) a larger problem into smaller sub-problems with teacher guidance or independently. |
| 1A-A-3-6 | Categorize a group of items based on the attributes or actions of each item, with or without a computing device. |
| 1A-A-3-7 | Construct and execute algorithms (sets of step-by-step instructions) that include sequencing and simple loops to accomplish a task, both independently and collaboratively, with or without a computing device. |
| 1A-A-6-8 | Analyze and debug (fix) an algorithm that includes sequencing and simple loops, with or without a computing device. |
| 1A-C-7-9 | Identify and use software that controls computational devices (e.g., use an app to draw on the screen, use software to write a story or control robots). |
| 1A-C-7-10 | Use appropriate terminology in naming and describing the function of common computing devices and components (e.g., desktop computer, laptop computer, tablet device, monitor, keyboard, mouse, printer). |
| 1A-C-6-11 | Identify, using accurate terminology, simple hardware and software problems that may occur during use (e.g., app or program not working as expected, no sound, device won't turn on). |
| 1A-D-7-12 | Collect data over time and organize it in a chart or graph in order to make a prediction. |
| 1A-D-4-13 | Use a computing device to store, search, retrieve, modify, and delete information and define the information stored as data. |
| 1A-D-4-14 | Create a model of an object or process in order to identify patterns and essential elements (e.g., water cycle, butterfly life cycle, seasonal weather patterns). |
| 1A-I-7-15 | Compare and contrast examples of how computing technology has changed and improved the way people live, work, and interact. |
| 1A-N-2-16 | Use computers or other computing devices to connect with people using a network (e.g., the Internet) to communicate, access, and share information as a class. |
| 1A-N-7-17 | Use passwords to protect private information and discuss the effects of password misuse |

| 3–5 | **Level 1B** |
| --- | --- |
| 1B-A-2-1 | Apply collaboration strategies to support problem solving within the design cycle of a program. |
| 1B-A-7-2 | Use proper citations and document when ideas are borrowed and changed for their own use (e.g., using pictures created by others, using music created by others, remixing programming projects). |
| 1B-A-5-3 | Create a plan as part of the iterative design process, both independently and with diverse collaborative teams (e.g., storyboard, flowchart, pseudo-code, story map). |
| 1B-A-5-4 | Construct programs, in order to solve a problem or for creative expression, that include sequencing, events, loops, conditionals, parallelism, and variables, using a block-based visual programming language or text-based language, both independently and collaboratively (e.g., pair programming). |
| 1B-A-5-5 | Use mathematical operations to change a value stored in a variable. |
| 1B-A-3-6 | Decompose (break down) a larger problem into smaller sub-problems, independently or in a collaborative group. |
| 1B-A-3-7 | Construct and execute an algorithm (set of step-by-step instructions) that includes sequencing, loops, and conditionals to accomplish a task, both independently and collaboratively, with or without a computing device. |
| 1B-A-6-8 | Analyze and debug (fix) an algorithm that includes sequencing, events, loops, conditionals, parallelism, and variables. |
| 1B-C-7-9 | Model how a computer system works. [Clarification: Only includes basic elements of a computer system, such as input, output, processor, sensors, and storage.] |
| 1B-C-7-10 | Use appropriate terminology in naming internal and external components of computing devices and describing their relationships, capabilities, and limitations. |
| 1B-C-6-11 | Identify, using accurate terminology, simple hardware and software problems that may occur during use, and apply strategies for solving problems (e.g., reboot device, check for power, check network availability, close and reopen app). |
| 1B-D-5-12 | Create a computational artifact to model the attributes and behaviors associated with a concept (e.g., solar system, life cycle of a plant). |
| 1B-D-5-13 | Answer a question by using a computer to (e.g., sort, total and/or average, chart, graph) and analyze data that has been collected by the class or student. |
| 1B-D-4-14 | Use numeric values to represent non-numeric ideas in the computer (binary, ASCII, pixel attributes such as RGB). |
| 1B-I-7-15 | Evaluate and describe the positive and negative impacts of the pervasiveness of computers and computing in daily life (e.g., downloading videos and audio files, electronic appliances, wireless Internet, mobile computing devices, GPS systems, wearable computing). |
| 1B-I-7-16 | Generate examples of how computing can affect society, and also how societal values can shape computing choices. |
| 1B-I-1-17 | Seek out and compare diverse perspectives, synchronously or asynchronously, to improve a project. |
| 1B-I-1-18 | Brainstorm ways in which computing devices could be made more accessible to all users. |
| 1B-I-1-19 | Explain problems that relate to using computing devices and networks (e.g., logging out to deter others from using your account, cyberbullying, privacy of personal information, and ownership). |
| 1B-N-7-20 | Create examples of strong passwords, explain why strong passwords should be used, and demonstrate proper use and protection of personal passwords. |
| 1B-N-4-21 | Model how a device on a network sends a message from one device (sender) to another (receiver) while following specific rules. |

| 6–8 | **Level 2** |
| --- | --- |
| 2-A-2-1 | Solicit and integrate peer feedback as appropriate to develop or refine a program. |
| 2-A-7-2 | Compare different algorithms that may be used to solve the same problem (e.g., different algorithms solve the same problem, but one might be faster than the other). [Clarification: Students are not expected to quantify these differences.] |
| 2-A-7-3 | Provide proper attribution when code is borrowed or built upon. |
| 2-A-7-4 | Interpret the flow of execution of algorithms and predict their outcomes. [Clarification: Algorithms can be expressed using natural language, flow and control diagrams, comments within code, and pseudocode.] |
| 2-A-5-5 | Design, develop, and present computational artifacts such as mobile applications that address social problems both independently and collaboratively. |
| 2-A-5-6 | Develop programs, both independently and collaboratively, that include sequences with nested loops and multiple branches. [Clarification: At this level, students may use block-based and/or text-based programming languages.] |
| 2-A-5-7 | Create variables that represent different types of data and manipulate their values. |
| 2-A-4-8 | Define and use procedures that hide the complexity of a task and can be reused to solve similar tasks. [Clarification: Students use and modify, but do not necessarily create, procedures with parameters.] |
| 2-A-3-9 | Decompose a problem into parts and create solutions for each part. |
| 2-A-6-10 | Use an iterative design process (e.g., define the problem, generate ideas, build, test, and improve solutions) to solve problems, both independently and collaboratively. |
| 2-C-7-11 | Justify the hardware and software chosen to accomplish a task (e.g., comparison of the features of a tablet vs. desktop, selecting which sensors and platform to use in building a robot or developing a mobile app). |
| 2-C-4-12 | Analyze the relationship between a device’s computational components and its capabilities. [Clarification: Computing Systems include not only computers, but also cars, microwaves, smartphones, traffic lights, and flash drives.] |
| 2-C-6-13 | Use a systematic process to identify the source of a problem within individual and connected devices (e.g., follow a troubleshooting flow diagram, make changes to software to see if hardware will work, restart device, check connections, swap in working components). |
| 2-D-7-14 | Describe how different formats of stored data represent tradeoffs between quality and size. [Clarification: compare examples of music, text and/or image formats.] |
| 2-D-7-15 | Explain the processes used to collect, transform, and analyze data to solve a problem using computational tools (e.g., use an app or spreadsheet form to collect data, decide which data to use or ignore, and choose a visualization method.). |
| 2-D-5-16 | Revise computational models to more accurately reflect real-world systems (e.g., ecosystems, epidemics, spread of ideas). |
| 2-D-4-17 | Represent data using different encoding schemes (e.g., binary, Unicode, Morse code, shorthand, student-created codes). |
| 2-I-7-18 | Summarize negative and positive impacts of using data and information to categorize people, predict behavior, and make recommendations based on those predictions (e.g., customizing search results or targeted advertising, based on previous browsing history, can save search time and limit options at the same time). |
| 2-I-7-19 | Explain how computer science fosters innovation and enhances nearly all careers and disciplines. |
| 2-I-1-20 | Provide examples of how computational artifacts and devices impact health and wellbeing, both positively and negatively. |
| 2-I-1-21 | Describe ways in which the Internet impacts global communication and collaborating. |
| 2-I-1-22 | Describe ethical issues that relate to computing devices and networks (e.g., equity of access, security and plagiarism), hacking, intellectual property, copyright, Creative Commons licensing. |
| 2-I-6-23 | Redesign a computational artifact to remove barriers to universal access (e.g., using captions on images, high contrast colors, and/or larger font sizes). |
| 2-N-7-24 | Summarize security risks associated with weak passwords, lack of encryption, insecure transactions, and persistence of data. |
| 2-N-4-25 | Simulate how information is transmitted as packets through multiple devices over the Internet and Networks. |

| 9–10 | **Level 3A** | **GDP 1** | | | | | | **GDP 2** | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **0** | **1** | **2** | **3** | **4** | **5** |
| 3A-A-2-1 | Design and develop a software artifact working in a team. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-A-2-2 | Demonstrate how diverse collaborating impacts the design and development of software products (e.g., discussing real-world examples of products that have been improved through having a diverse design team or reflecting on their own team's development experience). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-A-7-3 | Compare and contrast various software licensing schemes (e.g., open source, freeware, commercial). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-A-5-4 | Design, develop, and implement a computing artifact that responds to an event (e.g., robot that responds to a sensor, mobile app that responds to a text message, sprite that responds to a broadcast). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-A-5-5 | Use user-centered research and design techniques (e.g., surveys, interviews) to create software solutions |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-A-5-6 | Integrate grade-level appropriate mathematical techniques, concepts, and processes in the creation of computing artifacts. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-A-4-7 | Understand the notion of hierarchy and abstraction in high-level languages, translation, instruction sets, and logic circuits. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-A-4-8 | Deconstruct a complex problem into simpler parts using predefined constructs (e.g., functions and parameters and/or classes). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-A-4-9 | Demonstrate the value of abstraction for managing problem complexity (e.g., using a list instead of discrete variables). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-A-3-10 | Design algorithms using sequence, selection, and iteration. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-A-3-11 | Explain and demonstrate how modeling and simulation can be used to explore natural phenomena (e.g., flocking behaviors, queueing, life cycles). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-A-6-12 | Use a systematic approach and debugging tools to independently debug a program (e.g., setting breakpoints, inspecting variables with a debugger). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-C-7-13 | Develop and apply criteria (e.g., power consumption, processing speed, storage space, battery life, cost, operating system) for evaluating a computer system for a given purpose (e.g., system specification needed to run a game, web browsing, graphic design or video editing). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-C-5-14 | Create, extend, or modify existing programs to add new features and behaviors using different forms of inputs and outputs (e.g., inputs such as sensors, mouse clicks, data sets; outputs such as text, graphics, sounds). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-C-4-15 | Demonstrate the role and interaction of a computer embedded within a physical system, such as a consumer electronic, biological system, or vehicle, by creating a diagram, model, simulation, or prototype. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-C-4-16 | Describe the steps necessary for a computer to execute high compilation to machine language, interpretation, fetch-decode-execute <https://www.cise.ufl.edu/~mssz/CompOrg/CDAintro.html>. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-D-5-17 | Create computational models that simulate real-world systems (e.g., ecosystems, epidemics, spread of ideas). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-D-4-18 | Convert between binary, decimal, and hexadecimal representations of data (e.g., convert hexadecimal color codes to decimal percentages, ASCII/Unicode representation). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-D-4-19 | Analyze the representation tradeoffs among various forms of digital information (e.g., lossy versus lossless compression, encrypted vs. unencrypted, various image representations). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-D-3-20 | Discuss techniques used to store, process, and retrieve different amounts of information (e.g., files, databases, data warehouses). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-D-3-21 | Apply basic techniques for locating and collecting small- and large-scale data sets (e.g., creating and distributing user surveys, accessing real-world data sets). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-I-2-22 | Debate the social and economic implications associated with ethical and unethical computing practices (e.g., intellectual property rights, hacktivism, software piracy, diesel emissions testing scandal, new computers shipped with malware). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-I-7-23 | Compare and contrast information access and distribution rights. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-I-7-24 | Discuss implications of the collection and large-scale analysis of information about individuals (e.g., how businesses, social media, and government collect and use personal data). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-I-7-25 | Describe how computation shares features with art and music by translating human intention into an artifact. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-I-1-26 | Compare and debate the positive and negative impacts of computing on behavior and culture (e.g., evolution from hitchhiking to ridesharing apps, online accommodation rental services). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-I-1-27 | Demonstrate how computing enables new forms of experience, expression, communication, and collaborating. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-I-1-28 | Explain the impact of the digital divide (i.e., uneven access to computing, computing education, and interfaces) on access to critical information. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-I-6-29 | Redesign user interfaces (e.g., webpages, mobile applications, animations) to be more inclusive, accessible, and minimizing the impact of the designer's inherent bias. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-N-7-30 | Describe key protocols and underlying processes of Internet-based services (e.g., http/https and SMTP/IMAP, routing protocols). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-N-4-31 | Illustrate the basic components of computer networks (e.g., draw logical and topological diagrams of networks including routers, switches, servers, and end user devices; create model with string and paper). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-N-1-32 | Compare and contrast multiple viewpoints on cybersecurity (e.g., from the perspective of security experts, privacy advocates, the government). |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-N-3-33 | Explain the principles of information security (confidentiality, integrity, availability) and authentication techniques. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-N-3-34 | Use simple encryption and decryption algorithms to transmit/receive an encrypted message. |  |  |  |  |  |  |  |  |  |  |  |  |
| 3A-N-6-35 | Identify digital and physical strategies to secure networks and discuss the tradeoffs between ease of access and need for security. |  |  |  |  |  |  |  |  |  |  |  |  |

| 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. |
| 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). |
| 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). |