CENTER FOR CURRICULUM REDESIGN

# CCR Computer Science Benchmarks (Grades K-2)

| Standards | Benchmarks* |
|---|---|
| **CCR-CS CT 1. Exhibit computational thinking.** | CCR-CS CT K-2, 1.1 Use technology resources (e.g., puzzles, logical thinking programs) to solve age appropriate problems. [US-CSTA CT K-3, 1.1.] |
| | CCR-CS CT K-2, 1.2. Use writing tools, digital cameras, and drawing tools to illustrate thoughts, ideas, and stories in a step-by-step manner. [US-CSTA CT K-3, 1.2.] |
| | CCR-CS CT K-2, 1.3. Understand how to arrange (sort) information into useful order, such as sorting students by birth date, |
| | CCR-CS CT K-2, 1.4. Recognize that software is a set of ordered steps created to control computer operations. [US-CSTA CT K-3, 1.4.] |
| | |
| **CCR-CS CPP 2. Know computer programming.** | CCR-CS CPP K-2, 2.1. Construct a set of statements to be acted out to accomplish a simple task (e.g., Turtle instructions). [US-CSTA CPP K-3, 3.4.] |
| | CCR-CS CPP K-2, 2.2. Gather and organize information using concept-mapping tools, including flow charting. [US-CSTA |
| | CCR-CS CPP K-2, 2.3. Construct simple programs using programming languages and tools for this age group (e.g., |
| | CCR-CS CPP K-2, 2.4. Recognize safe and unsafe online behaviors (e.g., identifying which information should or should not |
| | |
| **CCR-CS CD 3. Understand computer hardware and communication systems.** | CCR-CS CD K-2, 3.1. Use standard input and output devices to successfully operate computers and related technologies. [US-CSTA CD K-3, 4.1.] |
| | CCR-CS CD K-2, 3.2. Recognize that computers are devices that execute programs. [US-CSTA CD 6-9 [K-3], 4.1.] |
| | CCR-CS CD K-2, 3.3. Demonstrate an appropriate level of proficiency with input and output devices. [US-CSTA CD 3-6 [K-3], |
| | CCR-CS CD K-2, 3.4. Understand the pervasiveness of computers and computing in daily life (e.g., voice mail, downloading |
| | CCR-CS CD K-2, 3.5. Understand that information is coming to and from a computer from many sources over a network |
| | CCR-CS CD K-2, 3.6. Identify factors that distinguish humans from machines. [US-CSTA CD 3-6 [K-3], 4.5.] |
| | CCR-CS CD K-2, 3.7. Recognize that computers are devices that execute programs. [US-CSTA CD 6-9 [K-3], 4.1.] |

*CSTA source noted in brackets is 2011 version of national standards

# CCR Computer Science Benchmarks (Grades 3-5)

| Standards | Benchmarks* |
|---|---|
| **CCR-CS CT 1. Exhibit computational thinking.** | CCR-CS CT 3-5, 1.1. Understand and use the basic steps in algorithmic problem-solving (e.g., problem statement and exploration, examination of sample instances, design, implementation, and testing). [US-CSTA CT 3-6, 1.1.] |
| | CCR-CS CT 3-5, 1.2. Develop a simple understanding of an algorithm (e.g., search, sequence of events, or sorting) using computer-free exercises. [US-CSTA CT 3-6, 1.2.] |
| | CCR-CS CT 3-5, 1.3. Demonstrate how a string of bits can be used to represent alphanumeric information. [US-CSTA CT 3-6, 1.3.] |
| | CCR-CS CT 3-5, 1.4. Describe how a simulation can be used to solve a problem. [US-CSTA CT 3-6, 1.4.] |
| | CCR-CS CT 3-5, 1.5. Make a list of sub-problems to consider while addressing a larger problem. [US-CSTA CT 3-6, 1.5.] |
| | |
| **CCR-CS CPP 2. Know computer programming.** | CCR-CS CPP 3-5, 2.1. Construct a program as a set of step-by-step instructions to be acted out (e.g., make a peanut butter and jelly sandwich activity). [US-CSTA CPP 3-6, 3.5.] |
| | CCR-CS CPP 3-5, 2.2. Implement problem solutions using a block-based visual programming language. [US-CSTA CPP 3-6, 3.6.] |
| | CCR-CS CPP 3-5, 2.3. Demonstrate good practices in personal information security, using passwords, encryption, and secure transactions. [US-CSTA CPP 6-9 [3-5], 3.6.] |
| | CCR-CS CPP 3-5, 2.4. Demonstrate strategies for validating information found on the internet (e.g., checking a less respected source against a more respected source; sleuthing the provenance of a claim; locating evidence separate from |
| | |
| **CCR-CS CD 3. Understand computer hardware and communication systems.** | CCR-CS CD 3-5, 3.1. Apply strategies for identifying and solving simple hardware and software problems that may occur during use (e.g., open device manager to see which program is malfunctioning, reboot the computer, check network |
| | CCR-CS CD 3-5, 3.2. Identify the physical location of documents, pictures, website, and programs that are interact with (e.g., on the local computer, on a commercial company's computer local network or in the cloud, on an attached drive). |
| | CCR-CS CD 3-5, 3.3. Understand the pathways by which information is transferred during routine operations (e.g., when backing up a file to an external drive, the information goes from the computer through the USB cord to a drive). |
| | CCR-CS CD 3-5, 3.4. Identify a variety of electronic devices that contain computational processors. [US-CSTA CD 6-9 [3-5], 4.2.] |
| | CCR-CS CD 3-5, 3.5. Describe the major components, functions, and organization of computer systems and networks (e.g., memory, software, operating system, input, output, processor, storage). [US-CSTA CD 6-9 [3-5], 4.6.] |
| | CCR-CS CD 3-5, 3.6. Recognize that computers model intelligent behavior as found in robotics, speech and language recognition, and computer animation. [US-CSTA CD 3-6, 4.6.] |

*CSTA source noted in brackets is 2011 version of national standards

# CCR Computer Science Benchmarks (Grades 6-8)

| Standards | Benchmarks* |
|---|---|
| **CCR-CS CT 1.**<br>**Exhibit computational thinking.** | CCR-CS CT 6-8, 1.1. Demonstrate and carry out the major processes and skills of exhibiting computational thinking as instructed in the grades 3-5 benchmarks with relative ease and automaticity. |
| | CCR-CS CT 6-8, 1.2. Describe and analyze a sequence of instructions being followed (e.g., describe a character's behavior in a video game as driven by rules and algorithms). [US-CSTA CT 6-9, 1.6.] |
| | CCR-CS CT 6-8, 1.3. Write an algorithm as a sequence of instructions that can be processed by a computer. [US- CSTA CT 6-9, 1.3.] |
| | CCR-CS CT 6-8, 1.4. Act out different searching and sorting algorithms. [US-CSTA CT 6-9, 1.5.] |
| | CCR-CS CT 6-8, 1.5. Describe advantages and disadvantages associated with different algorithms used to solve the same problem. [US-CSTA CT 6-9, 1.4. (Modified)] |
| | CCR-CS CT 6-8, 1.6. Use visual representations of problem states, structures, and data (e.g., graphs, charts, network diagrams, flowcharts). [US-CSTA CT 6-9, 1.8.] |
| | CCR-CS CT 6-8, 1.7. Understand the notion of hierarchy and abstraction in computing including high level languages, translation, instruction set, and logic circuits. [US-CSTA CT 6-9, 1.13.] |
| | CCR-CS CT 6-8, 1.8. Use abstraction to decompose a problem into sub problems. [US-CSTA CT 6-9, 1.12.] |
| | CCR-CS CT 6-8, 1.9. Use the basic steps in algorithmic problem solving to design solutions (e.g., problem statement and exploration, examination of sample instances, design, implementing a solution, testing, evaluation). [US-CSTA CT 6-9, 1.1.] |
| | CCR-CS CT 6-8, 1.10. Describe the process of parallelization as it relates to problem solving. [US-CSTA CT 6-9, 1.2.] |
| | CCR-CS CT 6-8, 1.11. Describe binary and hexadecimal numbers, and convert positive integers among decimal, binary, and hexadecimal number systems. |
| | CCR-CS CT 6-8, 1.12. Compare binary and hexadecimal representation of addresses and data (e.g., absolute addressing, character codes, colors). [CAN-ONT CS 11 [6-8], a5.2.] |
| | CCR-CS CT 6-8, 1.13. Design simple logic circuits using and, or, not, nand, and nor. [SG CA 10-11 [6-8], 2.1.4 (Modified)] |
| | CCR-CS CT 6-8, 1.14. Derive the truth tables of the fundamental logic gates (e.g., and, or, not, nor, nand, xor). [CAN-ONT CS 10 [6-8], a3.3.] |
| | CCR-CS CT 6-8, 1.15. Draw, design, and build simple logic circuits (e.g., adder circuit, decoder circuit) using logic chips and their truth tables, then validate the circuit's performance using a multimeter, logic probe, or other test equipment. [CAN-ONT CS 12 [6-8], a5.3. (Modified)] |
| | CCR-CS CT 6-8, 1.16. Examine connections between elements of mathematics and computer science including binary numbers, logic, sets and functions. [US-CSTA CT 6-9, 1.14.] |
| | CCR-CS CT 6-8, 1.17. Represent data in a variety of ways including text, sounds, pictures, and numbers. [US-CSTA CT 6-9, 1.7.] |

*CSTA source noted in brackets is 2011 version of national standards

| | |
|---|---|
| | CCR-CS CT 6-8, 1.18. Interact with content-specific models and simulations (e.g., ecosystems, epidemics, molecular dynamics) to support learning and research. [US-CSTA CT 6-9, 1.9.] |
| | CCR-CS CT 6-8, 1.19. Use different approaches to visualizing and analyzing small-to-medium-sized datasets. [CAN-ONT CS 11 [6-8], a5.1.] |
| | CCR-CS CT 6-8, 1.20. Analyze the degree to which a computer model accurately represents the real world. [US- CSTA CT 6-9, 1.11.] |
| | CCR-CS CT 6-8, 1.21. Evaluate what kinds of problems can be solved using modeling and simulation. [US-CSTA CT 6-9, 1.10.] |
| | |
| **CCR-CS CPP 2.**<br>**Know computer programming.** | CCR-CS CPP 6-8, 2.1. Demonstrate and carry out the major processes and skills of knowing computer programming as instructed in the grades 3-5 benchmarks with relative ease and automaticity. |
| | CCR-CS CPP 6-8, 2.2. Implement problem solutions using a programming language, including looping behavior, conditional statements, logic, expressions, variables, and functions. [US-CSTA CPP 6-9, 3.5.] |
| | CCR-CS CPP 6-8, 2.3. Collect and analyze the output from multiple runs of a computer program under different conditions (e.g., different configurations, different ranges of inputs). [US-CSTA CPP 6-9, 3.9.] |
| | CCR-CS CPP 6-8, 2.4. Use code from an open source project. |
| | |
| **CCR-CS CD 3.**<br>**Understand computer hardware and communication systems.** | CCR-CS CD 6-8, 3.1. Demonstrate and carry out the major processes and skills of understanding computer hardware and communication systems as instructed in the grades 3-5 benchmarks with relative ease and automaticity. |
| | CCR-CS CD 6-8, 3.2. Communicate about various technology, devices, and uses using developmentally appropriate, accurate terminology. [US-CSTA CD 6-9, 4.4.] |
| | CCR-CS CD 6-8, 3.3. Analyze and describe the unique features of computers embedded in mobile devices and vehicles (e.g., cell phones, automobiles, airplanes). [US-CSTA CD 9-12 [6-8], 4.1.] |
| | CCR-CS CD 6-8, 3.4. Explain the function of a communication system and the role of its components, including a source, encoder, transmitter, receiver, decoder, and storage. [7.MS-ETS3-1(MA).] |
| | CCR-CS CD 6-8, 3.5. Analyze and apply the strategies for identifying and solving more complex hardware and software problems that occur during everyday computer use (e.g., viruses, malware, hard drives needing defragmentation, file format incompatibility, managing file size limitations through file format tradeoffs). [US-CSTA CD 6-9, 4.5.] |
| | CCR-CS CD 6-8, 3.6. Describe ways in which computers use models of intelligent behavior (e.g., robot motion, speech and language understanding, and computer vision). [US-CSTA CD 6-9, 4.8.] |
| | CCR-CS CD 6-8, 3.7. Identify and explain how information is transferred from human to human, human to machine, machine to human, and machine to machine. [US-ITEEA ICT 6-8, 17-H (Modified).] |
| | CCR-CS CD 6-8, 3.8. Describe what distinguishes humans from machines focusing on human intelligence versus machine intelligence and ways to communicate through the use of self-learning, robotics, and computer animation. [US-CSTA CD 6-9, 4.7 and US-CSTA CD 3-6, 4.6.] |

CENTER FOR CURRICULUM REDESIGN

# CCR Computer Science Benchmarks (Grades 9-12)

| Standards | Benchmarks* |
|---|---|
| **CCR-CS CT 1. Exhibit computational thinking.** | CCR-CS CT 9-12, 1.1. Demonstrate and carry out the major processes and skills of exhibiting computational thinking as instructed in the grades 6-8 benchmarks with relative ease and automaticity. |
| | CCR-CS CT 9-12, 1.2. Describe a software development process used to solve software problems (e.g., design, coding, testing, verification). [US-CSTA CT 9-12, 1.2.] |
| | CCR-CS CT 9-12, 1.3. Compare and contrast simple data structures and their uses (e.g., arrays and lists). [US-CSTA CT 9-12, 1.17] |
| | CCR-CS CT 9-12, 1.4. Analyze the representation and trade-offs among various data structures (e.g., array, list, map, tree). [US-CSTA CT 9-12, 1.6.] |
| | CCR-CS CT 9-12, 1.5. Explain how sequence, selection, iteration, and recursion are building blocks of algorithms. [US-CSTA CT 9-12, 1.3.] |
| | CCR-CS CT 9-12, 1.6. Explain the value of abstraction to manage problem complexity. [US-CSTA CT 9-12, 1.9.] |
| | CCR-CS CT 9-12, 1.7. Use predefined functions, parameters, classes, and methods to divide a complex problem into simpler parts. [US-CSTA CT 9-12, 1.1.] |
| | CCR-CS CT 9-12, 1.8. Decompose a problem by defining new functions and classes. [US-CSTA CT 9-12, 1.21.] |
| | CCR-CS CT 9-12, 1.9. Evaluate algorithms by their efficiency, correctness, and clarity. [US-CSTA CT 9-12, 1.15.] |
| | CCR-CS CT 9-12, 1.10. Evaluate tradeoffs associated with different algorithms used to solve the same problem. [US- CSTA CT 6-9 [9-12], 1.4.] |
| | CCR-CS CT 9-12, 1.11. Critically examine classical algorithms and implement an original algorithm. [US-CSTA CT 9- 12, 1.14.] |
| | CCR-CS CT 9-12, 1.12. Classify problems as tractable, intractable, or computationally unsolvable. [US-CSTA CT 9- 12, 1.12.] |
| | CCR-CS CT 9-12, 1.13. Explain the value of heuristic algorithms to approximate solutions for intractable problems (e. g., traveling salesman problem). [US-CSTA CT 9-12, 1.13.] |
| | CCR-CS CT 9-12, 1.14. Describe the concept of parallel processing as a strategy to solve large problems. [US-CSTA CT 9-12, 1.10.] |
| | CCR-CS CT 9-12, 1.15. Demonstrate concurrency by separating processes into threads and dividing data into parallel streams. [US-CSTA CT 9-12, 1.22.] |
| | CCR-CS CT 9-12, 1.16. Explain how the binary number system and binary coding are used to represent a variety of forms (e.g., instructions, numbers, text, sound, images) using bits and bytes. [US-CSTA CT 9-12, 1.18 (modified)] |
| | CCR-CS CT 9-12, 1.17. Use binary coding to represent a variety of forms (e.g., instructions, numbers, text, sound, images). |
| | CCR-CS CT 9-12, 1.18. Explain how to interpret a binary sequence that represents a specific form (e.g., instructions, numbers, text, sound, image). [US-CSTA CT 9-12, 1.18 (modified)] |

*CSTA source noted in brackets is 2011 version of national standards

| | |
|---|---|
| | CCR-CS CT 9-12, 1.19. Use data analysis to enhance understanding of complex natural and human systems. [US- CSTA CT 9-12, 1.16] |
| | CCR-CS CT 9-12, 1.20. Analyze data and identify patterns through modeling and simulation. [US-CSTA CT 9-12, 1.] |
| | CCR-CS CT 9-12, 1.21. Use or modify modeling and simulation software to represent and understand natural phenomena. [US-CSTA CT 9-12, 1.8.] |
| | CCR-CS CT 9-12, 1.22. Use models and simulations to help formulate, refine, and test problem solutions (e.g., scientific hypotheses of natural phenomena, analyze existing systems, test proposed systems, recognize strengths and limitations of design solutions). [US-CSTA CT 9-12, 1.19 (modified)] |
| | CCR-CS CT 9-12, 1.23. Use different approaches to visualizing and analyzing massive data sets. [US-CSTA CT 9- 12, 1.4.] |
| | |
| **CCR-CS CPP 2.**<br>**Know computer programming.** | CCR-CS CPP 9-12, 2.1. Demonstrate and carry out the major processes and skills of knowing computer programming as instructed in the grades 6-8 benchmarks with relative ease and automaticity. |
| | CCR-CS CPP 9-12, 2.2. Describe a variety of programming languages available to solve problems and develop systems. [US-CSTA CPP 9-12, 3.7.] |
| | CCR-CS CPP 9-12, 2.3. Classify programming languages based on their level and application domain. [US-CSTA CPP 9-12, 3.15.] |
| | CCR-CS CPP 9-12, 2.4. Explain the program execution process. [US-CSTA CPP 9-12, 3.8.] |
| | CCR-CS CPP 9-12, 2.5. Demonstrate the software life cycle process by participating on a software project team (e. g., tests, pull requests, merging). [US-CSTA CL 9-12, 2.6.] |
| | CCR-CS CPP 9-12, 2.6. Participate in and contribute to an open source project. |
| | CCR-CS CPP 9-12, 2.7. Apply analysis, design, and implementation techniques to solve problems (e.g., use one or more software lifecycle models). [US-CSTA CPP 9-12, 3.4.] |
| | CCR-CS CPP 9-12, 2.8. Use project collaboration tools, version control systems (e.g., Github), and integrated development environments (IDE). [US-CSTA CL 9-12, 2.5.] |
| | CCR-CS CPP 9-12, 2.9. Select appropriate file formats for various types and uses of data. [US-CSTA CPP 9-12, 3.6.] |
| | CCR-CS CPP 9-12, 2.10. Use application program interfaces (APIs) and libraries to facilitate programming solutions. [US-CSTA CPP 9-12, 3.5.] |
| | CCR-CS CPP 9-12, 2.11. Use tools of abstraction to decompose a large-scale computational problem (e.g., procedural abstraction, object-oriented design, functional design). [US-CSTA CPP 9-12, 3.14.] |
| | CCR-CS CPP 9-12, 2.12. Use various debugging and testing methods to ensure program correctness (e.g., test cases, unit testing, white box, black box, integration testing). [US-CSTA CPP 9-12, 3.3.] |
| | CCR-CS CPP 9-12, 2.13. Evaluate programs written by others for readability and usability (e.g., code reviews). [US- CSTA CL 9-12, 2.7.] |

| | |
|---|---|
| | CCR-CS CPP 9-12, 2.14. Explain and use the principles of security by examining encryption, cryptography, and authentication techniques. [US-CSTA CPP 9-12, 3.9.] |
| | CCR-CS CPP 9-12, 2.15. Deploy principles of security by implementing encryption and authentication strategies. [US-CSTA CPP 9-12, 3.17.] |
| | CCR-CS CPP 9-12, 2.16. Demonstrate an understanding of laws and regulations surrounding data privacy (e.g., HIPAA, COPPA, FERPA in the US; in the EU, the Data Protection Directive) along with related tradeoffs. |
| | CCR-CS CPP 9-12, 2.17. Describe and use techniques for locating and collecting small and large-scale data sets. [US-CSTA CPP 9-12, 3.11.] |
| | CCR-CS CPP 9-12, 2.18. Describe how mathematical and statistical functions, sets, and logic are used in computation. [US-CSTA CPP 9-12, 3.12.] |
| | CCR-CS CPP 9-12, 2.19. Use advanced tools to create digital artifacts (e.g., web design, animation, video, multimedia). [US-CSTA CPP 9-12, 3.13.] |
| | CCR-CS CPP 9-12, 2.20. Create and organize web pages through the use of a variety of web programming design tools. [US-CSTA CPP 9-12, 3.1.] |
| | CCR-CS CPP 9-12, 2.21. Use mobile devices and emulators to design, develop, and implement mobile computing applications. [US-CSTA CPP 9-12, 3.2.] |
| | CCR-CS CPP 9-12, 2.22. Explore principles of system design in scaling, efficiency, and security. [US-CSTA CPP 9- 12, 3.16.] |
| | |
| **CCR-CS CD 3. Understand computer hardware and communication systems.** | CCR-CS CD 9-12, 3.1. Demonstrate and carry out the major processes and skills of understanding computer hardware and communication systems as instructed in the grades 6-8 benchmarks with relative ease and automaticity. |
| | CCR-CS CD 9-12 3.2. Identify and describe hardware (e.g., physical layers, logic gates, chips, components). [US- CSTA CD 9-12, 4.12.] |
| | CCR-CS CD 9-12 3.3. Explain the multiple levels of hardware and software that support program execution (e.g., compilers, interpreters, operating systems, networks). [US-CSTA CD 9-12, 4.5.] |
| | CCR-CS CD 9-12 3.4. Identify and select the most appropriate file format based on trade-offs (e.g., accuracy, speed, ease of manipulation). [US-CSTA CD 9-12, 4.13.] |
| | CCR-CS CD 9-12 3.5. Explain the basic components of computer networks (e.g., servers, file protection, routing, spoolers and queues, shared resources, and fault-tolerance). [US-CSTA CD 9-12, 4.8.] |
| | CCR-CS CD 9-12 3.6. Compare and contrast client-server and peer-to-peer network strategies and explain the difference between the two types. [US-CSTA CD 9-12, 4.7.] |
| | CCR-CS CD 9-12 3.7. Apply strategies for identifying and solving routine hardware and software problems that occur in everyday life, including problems with networked devices. [US-CSTA CD 9-12, 4.6.] |
| | CCR-CS CD 9-12 3.8. Identify and solve issues that impact network functionality (e.g., latency, bandwidth, firewalls, server capability). [US-CSTA CD 9-12, 4.14.] |
| | CCR-CS CD 9-12 3.9. Develop criteria for purchasing or upgrading computer system hardware. [US-CSTA CD 9-12, 4.2.] |

| | |
|---|---|
| | CCR-CS CD 9-12 3.10. Discuss the impact of specific modifications on the functionality of application programs and computer systems (i.e., foresee the specific ripple effects caused by small changes to a program or system). [US- CSTA CD 9-12, 4.11.] |
| | CCR-CS CD 9-12 3.11. Describe the major applications of artificial intelligence and robotics. [US-CSTA CD 9-12, 4.10.] |
| | CCR-CS CD 9-12 3.12. Explain the notion of intelligent behavior through computer modeling and robotics. [US-CSTA CD 9-12, 4.15.] |