CCR Computer Science Standards	CCR Computer Science Benchmarks (Grades 6-8)*
CCR-CS CT 1. 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.]
	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. 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. 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.]
	*CSTA source noted in brackets is 2011 version of national standards