CCR Computer Science Standards	CCR Computer Science Benchmarks (Grade 9-12)*
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)]
	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. 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.]

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