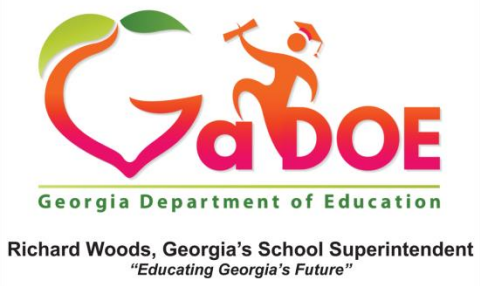
Georgia
Standards of Excellence

Computer Science

Standards

Middle School Computer Science II



Georgia Standards of Excellence for K-8 Computer Science

Georgia Standards of Excellence (GSE) for Computer Science (CS) were created in response to the growing ubiquity of computing devices and their impact on every aspect of society. If Georgia's students are to participate effectively in society, a shift in K-12 education must correspond. In Georgia, Computer Science is understood as the study of computers and algorithmic processes, including their principles, their hardware and software designs, their implementation, and their impact on society. The standards blend the core concepts of computer science (i.e., what students should know) and computer science practices (i.e., what students should do). These core concepts and practices should be taught in an integrated way to provide authentic learning experiences for students.

The GSE for Computer Science immerse students in the practices of Computer Science from Kindergarten through grade 12, effectively transitioning Computer Science from a high school elective to a comprehensive K-12 discipline for all students. Some skills or concepts are emphasized more in particular grade bands in conjunction with research on how students learn and other knowledge and skills taught at those levels. Any curriculum aligned to these GSE should revisit domains and concepts over time as students apply their learning by creating computational artifacts. Creating computational artifacts can be as simple as writing socially responsible electronic messages (e.g., email and social media posts) and as complex as designing an app for a drone or a self-driving vehicle.

The standards are organized in grade bands rather than grade levels to afford schools flexibility in presenting the content while maintaining a structured, developmental progression from one band to another. Teachers can scaffold instruction from simple familiarization in the K-2 grade band to deeper involvement in the 3-5 and more thorough treatment in the 6-8 grade band. In addition, the 6-8 grade band standards are designed to feed directly into the high school CS pathways which are, in turn, designed to meet the dynamic needs of industry and post-secondary study of computer science.

Georgia-owned and Georgia-grown, the GSE for Computer Science relate broadly to national and international frameworks. The grade bands follow the structure set forth by the K12 CS Framework; they develop a comprehensive conceptual framework that grows over the years. The K-8 GSE for Computer Science also correspond to the ISTE standards for students as organizational domains. These domains are intended to be cross-curricular. The ISTE domains (e.g. Empowered Learner) define a high-level perspective on the characteristics of a 21st century student. These characteristics are couched in a digital society but are not restricted to computer science content. Likewise, the GSE for Computer Science can be integrated into other content areas and support enduring characteristics for learning (e.g., collaborative, communicative, creative, and critical thinking). Ultimately, the GSE for Computer Science support and inspire Georgia's students as they grow and learn, empowering students to be successful, responsible, and engaged citizens.

Georgia Standards of Excellence for K-8 Computer Science

The Standards are written in the following format:

CSS = Computer Science Standard
EL = Empowered Learner (Domain)
6-8 = Grade band 6 through 8
1 = is the standard number
1… = Element of the standard

Cluster 6-8
Empowered Learner
CSS.EL.6-8.1
Use technology resources to increase self-direction and self-regulation in learning, including for problem solving and collaboration (e.g., using the Internet to access online resources, edit documents collaboratively)

1. Understand the difference between editing a shared document and suggesting edits (e.g. track changes)
2. Use digital tools or platforms to organize, display, annotate, and/or share a curated collection
3. Complete an individual project (e.g., research or design) using technology resources

Georgia Standards of Excellence for K-8 Computer Science

Table of Contents

Empowered Learner
CSS.EL.6-8.1
Use technology resources to increase self-direction and self-regulation in learning, including for problem solving and collaboration (e.g., using the Internet to access online resources, edit documents collaboratively).

1. Understand the difference between editing a shared document and suggesting edits (e.g. track changes).
2. Use digital tools or platforms to organize, display, annotate, and/or share a curated collection.
3. Complete an individual project (e.g., research or design) using technology resources.

Digital Citizen
CSS.DC.6-8.3
Explore computer science and computing-related careers.

1. Investigate a career that requires computing and technology.
2. Describe how computer science enhances other career fields.
3. Predict the role of computer science in future careers.

CSS.DC.6-8.8
Investigate and identify the basic components of computers and networks.

1. Identify the basic components of the computer by disassembling and reassembling a demonstration model personal computer (can be done 'virtually' online if demo model is not available).
2. Demonstrate an understanding of key functional components (input devices, output devices, processor, operating system, software applications, memory, storage, wi-fi and/or ethernet ports, and IP addresses).
3. Demonstrate an understanding of the terms and units used to describe major hardware components (RAM, ROM, GHz, MHz, GB, MB, CD, DVD, RW).
4. Explain the interrelation of the operating system software, application software, and utility software, citing specific examples of each.
5. Develop a basic vocabulary of networks including the Internet, wired, wireless, cellular, wi-fi, messages, packets, connections, bandwidth, broadband, firewall, hacking, cybersecurity, encryption, local area network (LAN), wide area network (WAN), and OSI model.
6. Demonstrate an understanding of the fundamental concepts for how computers process programming commands (hex, binary language, sequence of commands, conditional structures, looping structures).

Conceptual Category Networks and the Internet
CSS.DC.6-8.9
Investigate ways to differentiate networks and how they are used in business and industry.

1. Create diagrams to illustrate types of network topologies to include star, ring, bus, mesh, and hybrid.
2. Differentiate networks based on coverage area including local area network (LAN), wide area network (WAN), and personal area network (PAN)
3. Differentiate between different network mediums including Wi-fi, wired, satellite, and microwave.

CSS.DC.6-8.10
Evaluate and provide a rationale for the levels of the Open Systems Interconnection (OSI) model.

1. Summarize from multiple sources the physical and digital aspects of computing networks.
2. Trace the layers required to transmit data from one node to another (the OSI model).
3. Construct and explain the basic functions of the OSI model.

CSS.DC.6-8.11
Examine the basics of cybersecurity needs for business, government, and organizations.

1. List and define the elements of the confidentiality, integrity, and availability (CIA) triad.
2. Explain components of access control: Identification, Authentication, Authorization, Accountability, and Non-repudiation.
3. Identify the characteristics of strong vs. weak passwords in data and identity security.
4. List and describe the basic steps in security risk management.
5. Develop a logical argument for the importance of physical security.

CSS.DC.6-8.12
Cite evidence regarding the principles of cybersecurity and basic mechanisms used for protecting data and resources.

1. Define the cybersecurity first principles of least privilege, minimization, abstraction, domain separation, process isolation, information hiding, layering, simplicity, modularity, and resource encapsulation.
2. Apply concepts related to the principles behind encryption, including the purpose of cryptography, hashing, and steganography.
3. Draw conclusions illustrating a basic understanding of internet protocol (IP) packets, ports and network transmission.
4. Summarize from multiple sources a basic understanding of anti-malware, firewalls, intrusion detection system/intrusion prevention system (IDS/IPS), and virtual private network (VPN).

CSS.DC.6-8.13
Analyze and describe the characteristics of cybersecurity ethics, digital citizenship, and laws governing privacy.

1. Explain the differences between a white hat (ethical) hacker and a black hat (unethical) hacker.
2. Cite evidence regarding the practice of ethical digital decision-making, including plagiarism, copyright law, and software licensing types (freeware, public domain, shareware, etc.).
3. Summarize and provide examples regarding security and privacy laws and their impact on society, citing recent cases.
4. Analyze cyberbullying to include legal and social consequences
5. Develop a set of guidelines to prevent cyberbullying.
6. Develop arguments for policy-driven and technology-driven security.

Innovative Designer and Creator
CSS.IDC.6-8.21
Develop a plan to create, design, and build a website with digital content to a specific target market.

1. Identify the objectives (e.g., increase sales, promote new products, increase company awareness, target new customers) for the website's target market.
2. Specify website requirements, including timeline and resources, and organize them into a requirements document.
3. Find and evaluate similar websites (in terms of overall function and layout) using an evaluation instrument for side-by-side comparison. Consider major design elements (ease of use, responsiveness, adaptability to mobile, tablet and desktop, etc.).
4. Evaluate a variety of web design tools and development platforms using an evaluation instrument and choose the appropriate platform.

CSS.IDC.6-8.22
Design digital products that reveal a professional layout and look by applying design principles to produce professional quality digital products.

1. Identify graphical elements and the appropriate use of elements on a web site.
2. Explore and apply color principles to digital products.
3. Establish a brand through consistent use of graphics, color, layout and text.
4. Analyze the look and layout of a website based on the first impression of content and page elements. Get feedback from independent people and incorporate where appropriate.

CSS.IDC.6-8.23
Create a single functional web page using a web development platform based on a design mockup and user requirements.

1. Create and edit images and graphics for website publication.
2. Plan, produce, and edit digital audio for website publication.
3. Plan, produce, edit, and post a multimedia-rich video project to a website.
4. Plan, produce, and edit animations for website publication.

CSS.IDC.6-8.24
Develop and use a test plan to debug each new website version to ensure it runs as intended and meets the end-user requirements for a responsive site.

1. Create a test and debug plan. Resolve issues and fix any errors that surface during the test and debug process.
2. Create an end user testing plan, get user feedback, and incorporate feedback into the final website.
3. Prepare website for publishing and promotion.

CSS.IDC.6-8.25
Develop a plan to create, design, and build a game with digital content for a specific target market.

1. Explore various game types including role-playing games (RPG), real-time strategy (RTS), simulations, puzzles, educational, massively multiplayer online (MMO), and others.
2. Create a Game Design Document (GDD), which includes, characters, story, theme, and gameplay mechanics.

CSS.IDC.6-8.26
Develop a visual model of a game from the Game Design Document (GDD).

1. Create storyboards from the GDD that demonstrate game progression and consistent use of a theme.
2. Use the GDD to design the wireframes and comprehensive layout for the user experience (UX).

CSS.IDC.6-8.27
Create a functional game, using a game development platform, based on the storyboards, wireframes, and comprehensive layout.

1. Create game elements, backgrounds, and characters.
2. Use scripting languages to create desired game mechanics, and to control the environment, user interface (UI), and character behaviors.
3. Plan, produce, and edit graphics and animations for game publication.
4. Plan, produce, and edit digital audio for game publication.

CSS.IDC.6-8.28
Develop and use a test plan to debug use each time a version of the game is released to ensure it runs as intended and meets the end-user requirements.

1. Create a test and debug plan. Resolve any issues and fix any errors that surface during the test and debug process.
2. Create an end user testing plan, get user feedback, and incorporate feedback into the final game.
3. Prepare final game for publishing prior to publishing to the target audience.

Computational Thinker
Conceptual Category: Recognizing and Defining Computational Problems

CSS.CT.6-8.32
Develop through application, logical observations relative to computational thinking procedures to analyze and solve problems current to everyday life.

1. Identify characteristics of computational thinking (decomposition, pattern recognition, algorithmic thinking and abstraction).
2. Explain issues and analyze routine hardware and software problems current to everyday life.
3. Apply troubleshooting concepts to issues regarding compatibility, data, and identity.
4. Describe ways to resolve operational problems caused by hardware errors.
5. Explain how technology can create ethical and legal issues in the business world and a technology-based society and how it can be used to solve & manage those issues.

CSS.CT.6-8.33
Utilize computational thinking to solve problems.

1. Make observations and organize the concepts of modularity, including functions and methods, as it relates to programming code reusability and cloud computing in the software industry.
2. Develop a working vocabulary of computational thinking including sequences, algorithms, binary, pattern matching, decomposition, abstraction, parallelization, data, automation, data collection, data analysis, boolean, integer, branches (if...then...else), and iteration {loops (For, While)}.
3. Analyze the problem-solving process, the input-process-output-storage model of a computer, and how computers help humans solve problems.
4. Develop an algorithm to decompose a problem of a daily task.

CSS.CT.6-8.34
Recognize when to use the same solution for multiple problems.

Conceptual Category: Data and Information
CSS.CT.6-8.35
Evaluate the storage and representation of data; Analyze how data is collected with both computational and non-computational tools and processes.

1. Discuss binary numbers, logic, sets, and functions and their application to computer science.
2. Explain that searches may be enhanced by using Boolean logic (e.g., using “not”, “or”, “and”).

Conceptual Category: Algorithms
CSS.CT.6-8.36
Understand and use the basic steps in algorithmic problem solving in computing and other authentic applications.

1. Select basic steps to solve algorithmic problems.
2. Evaluate basic steps of algorithmic problem solving to design solutions.
3. Solve algorithmic problems of increasing complexity.

Conceptual Category: Programming
CSS.CT.6-8.37
Use and compare simple coding control structures (e.g., if-then, loops)

1. Use a visual block-based and/or text-based programming language individually and collaboratively to solve problems of increasing complexity.
2. Create a program individually and collaboratively using a text-based programming. language; Identify variables and compare the types of data stored as variables.

Conceptual Category: Creating Computational Artifacts
CSS.CT.6-8.38
Consider the purpose of computational artifacts for practical use, personal expression, and/or societal impact.

1. Compare and contrast examples of high level and low-level programming languages.
2. Investigate the notion of hierarchy in computing including high level languages, translations, instruction sets, and logic circuits.
3. Develop problem solutions using a programming language, including all of the following: looping behavior, conditional statements, expressions, variables, and functions.

Conceptual Category: Testing and Refining Computational Artifacts
CSS.CT.6-8.39
Test computational artifacts systematically by considering multiple scenarios and using test cases.

Conceptual Category: Human Computer Interaction
CSS.CT.6-8.40
Describe how humans and machines interact to accomplish tasks that cannot be accomplished by either alone.

1. Identify what distinguishes humans from machines focusing on human intelligence versus machine intelligence (e.g., robot motion, speech and language understanding, and computer vision); Explain why some tasks can be accomplished more easily by computers.
2. Describe ways in which computers use models of intelligent behavior (e.g., robot motion, speech and language understanding, and computer vision) and how they differ (e.g., emotional decision making versus logical decisions, common sense, literal versus abstract).
3. Design and demonstrate the use of a device (e.g., robot, e-textile) to accomplish a task, individually and collaboratively.

Creative Communicator
Conceptual Category: Collaborating Around Computing
CSS.CC.6-8.41
Use online resources to participate in collaborative activities for the purpose of developing solutions or products.

CSS.CC.6-8.42
Improve teamwork and collaboration skills: providing useful feedback, integrating feedback, understanding, and accepting multiple perspectives.

1. Understand the difference between CC and BCC as well as Reply and Reply All and when to use each appropriately.

CSS.CC.6-8.43
Collaborate productively and recognize the value of working with individuals of varying perspectives, skills, and backgrounds.

1. Set and implement equitable expectations and workloads when working in teams.

CSS.CC.6-8.44
Demonstrate correct keyboarding techniques while increasing speed and maintaining accuracy.

CSS.CC.6-8.45
Use productivity technology tools (e.g. word processing, spreadsheet, presentation software) for individual and collaborative writing, communication, and publishing activities.

Global Collaborator
Conceptual Category: Fostering an Inclusive Computing Culture
CSS.GC.6-8.46
Recognize that equitable access to computing benefits society as a whole.

CSS.GC.6-8.47
Consider others' perspectives as well as one's own perspective when developing computational solutions.

CSS.GC.6-8.48
Consider the needs of a variety of end users regarding accessibility and usability.

CSS.GC.6-8.49
Use software applications to collaborate and create authentic products.

1. Identify and utilize the appropriate software application for productivity.
2. Use various applications in a professional manner to share and communicate with peers and teachers.
3. Share documents created using word processing, presentation, and spreadsheet software.
4. Create original works using software applications in a collaborative manner.
5. Collaborate in small groups to create and edit online documents in real time.
6. Identify and use appropriate file sharing strategies (e.g., copy and paste, links, posts, and attachments).

Glossary of Computer Science Terms

These terms are used throughout the standards. They are content-specific vocabulary for Computer Science.

Term	Definition	Example
Abstraction	The process of taking away or removing characteristics from something in order to make it less complex. The product should be a new representation of essential characteristics. The new representation hides details that are irrelevant to the problem at hand.	To represent a person, an abstraction may include two arms, two legs, a head, and a torso but no hair or toes. This representation gives enough information to show a person without being too complex.
Algorithm	Detailed, step-by-step instructions for solving a problem or completing a task.	The set of steps used to solve a long division problem is an example of an algorithm.
Analog	A defining characteristic of data; analog data are stored in a continuous transmission of a signal. It is often contrasted with digital, which is how computers store and process data as a set of individual symbols.	A compact disc is digital; a vinyl record is analog.
Artifact (computational)	Anything created by a human using a computer.	A word processing document, an app, and a webpage are all computational artifacts.
Binary	A number system using only on the numerals 0 and 1.	The binary number 01011 converts to 11 in decimal numbers.
Biometric	The measurement and analysis of unique physical or behavioral characteristics (such as fingerprint or voice patterns) especially as a means of verifying personal identity.	Fingerprint scanners utilize a biometric evaluation to grant access.

Block-based programming language	A visual representation of common sets of instructions for coding that can be organized to create computer programs; block-based programing is often used to teach coding to younger or novice learners.	A popular block-based coding language is Blockly.
Coding	Creating a computer program.	Writing directions for a computer using a computer language such as Java, Python, or Blockly.
Computational Thinking	A problem-solving process used to formulate problems in a way that a computer and other tools could be used to help solve.	

Conditional	A programming statement, often starting with "if", in which one half expresses something that depends on the other half.	If student's grade is greater than or equal to 60
		Print "passed"
		else
		Print "failed"
		endif

Curate	Collect, organize, and present information typically using professional or expert knowledge.	Selecting a set of pictures to share or add to a photo album.
Debug	The process of finding and removing errors from computer programs.	Correcting errors.
Decomposition	Specific to computer science, decomposition means breaking a complex problem or system into parts that are easier understand.	To create an app that calculates an ideal heart rate, the program would break down the process to input of information from a patient, calculation of that information, and output of the ideal heart rate.
Design Process	A formal approach used by architects, engineers, and scientists for breaking down a large project into manageable chunks.	

Drone	A remote-controlled pilotless aircraft or missile.	A photographer can use a drone to take aerial pictures.
Event	An action or occurrence recognized by software, often originating from the external environment, that may be handled by the software.	Accepting input from a user is an event that may be followed up by some processing activity.
Hexadecimal (Hex)	Relating to or using a system of numerical notation that has 16 digits rather than 10 as its base.	The number 15 in our common base ten decimal system is represented with the letter 'F' in hexadecimal.
Ideate	The process of generating ideas and solutions.	Sketching, prototyping, or brainstorming can be processes for ideation.
Loop	A sequence of instructions that is continually repeated until a certain condition is reached.	An action that is performed again and again by a computer program.
Model	Constructing a representation of some part of a problem or system.	A budget is a model for how money is spent and earned.
Ordinal	Relating to an ordinal number; representing a position in a series.	1st, 2nd, 3rd, 4th, ...
Phishing	The fraudulent practice of sending emails purporting to be from reputable sources in order to entice individuals to reveal personal information, such as passwords and credit card numbers.	A phishing email is a fake message from a place like the Internal Revenue Service requesting a social security number.
Prototype	A model of something from which other forms are developed or copied.	At an auto show, a "concept car" is a prototype of a car that may go into mass production.
Remix	To change a set of code by adding or rearranging smaller code segments to create a different outcome.	A computer program that uses segments of other programs to solve a problem.

Scratch	A block-based programming platform commonly used for novice programmers.	
Sequence	An ordered, step-by-step process of an action or event proceeding in a pattern.	5, 10, 15, 20 is a sequence that relies on a pattern of +5.
Unplugged	Activities used for teaching computational thinking or computer science without a computing device.	Using playing cards to teach sorting is an unplugged activity to teach how computers sort data.