Coding in Minecraft Advanced Coding with JavaScript

Scope and Sequence - Georgia

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
Lesson 1: Introduction to MakeCode with Minecraft (Optional lesson) Average Completion time: Approximately 2 hours	• Learners will develop an understanding of how they can use code to build in Minecraft and develop a relationship with 'Agent.' • Learners code Agent to navigate through a maze. • Learners program Agent to place and break blocks • Learners will design and create a code (including a loop) to build a structure.	**CSS.KC.6-8.20** Design, develop, debug and implement computer programs. 3. Develop a working vocabulary of programming including flowcharting and/or storyboarding, coding, debugging, user interfaces, usability, variables, lists, loops, conditionals, programming language, events. **CSS.CT.6-8.37** Use and compare simple coding control structures 1. Use a visual block-based and/or text-based programming language individually and collaboratively to solve problems of increasing complexity.	**1B-AP-09** Create programs that use variables to store and modify data. **1B-AP-10** Create programs that include sequences, events, loops, and conditionals. **1B-AP-11** Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process. **1B-AP-13** Use an iterative process to plan the development of a program by including others' perspectives and considering user preferences.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		CSS.CT.3-5.5 Develop and employ strategies for understanding and solving problems in ways that use the power of technological methods to develop and test solutions. 2. Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process. 4. Create programs that include sequences, events, loops, conditionals, and variables. **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.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
Lesson 2: Writing Computer Algorithms Average Completion time: Approximately 2 hours	• Learners will understand what an algorithm is. • Learners will write algorithms. Learners will understand the requirement for thorough instructions in computer programs.	**CSS.KC.6-8.20** Design, develop, debug and implement computer programs. 1. Develop a working vocabulary of programming including flowcharting and/or storyboarding, coding, debugging, user interfaces, usability, variables, lists, loops, conditionals, programming language, events. **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.	**2-AP-10** Use flowcharts and/or pseudocode to address complex problems as algorithms. **2-AP-14** Create procedures with parameters to organize code and make it easier to reuse.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		CSS.CT.6-8.33 Utilize computational thinking to solve problems. 4. Develop an algorithm to decompose a problem of a daily task.	
Lesson 3: Moving to JavaScript Average Completion time: Approximately 2 hours	• Learners will Understanding text-based programming language • Learners will be able to identify equivalent JavaScript code for MakeCode blocks Learners will be able to make modifications to JavaScript code to affect the output	**CSS.KC.6-8.20** Design, develop, debug and implement computer programs. 1. Develop a working vocabulary of programming including flowcharting and/or storyboarding, coding, debugging, user interfaces, usability, variables, lists, loops, conditionals, programming language, events. 5. Implement a simple algorithm in a computer program. 6. Develop an event driven program.	**2-AP-12** Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. **2-AP-14** Create procedures with parameters to organize code and make it easier to reuse. **2-AP-17** Systematically test and refine programs using a range of test cases

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		CSS.CT.6-8.37 Use and compare simple coding control structures 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. **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.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		2. Evaluate basic steps of algorithmic problem solving to design solutions. 3. Solve algorithmic problems of increasing complexity. **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.	
Lesson 4: JavaScript Loops Average Completion time: Approximately 3 hours	• Learners will identify and modify Repeat loops in JavaScript • Learners will identify and modify While loops in JavaScript Learners will identify and modify For loops in JavaScript	**CSS.KC.6-8.20** Design, develop, debug and implement computer programs. 1. Develop a working vocabulary of programming including flowcharting and/or storyboarding, coding, debugging, user interfaces, usability, variables, lists,	• **2-AP-12** Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. • **2-AP-14** Create procedures with parameters to organize code and make it easier to reuse.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		loops, conditionals, programming language, events. 5. Implement a simple algorithm in a computer program. 6. Develop an event driven program. 7. Create a program that accepts user and/or sensor input and stores the result in a variable. 8. Create a computer program that implements a loop. 9. Develop a program that makes a decision based on data or user input. 10. Debug a program with an error. **CSS.CT.6-8.37** Use and compare simple coding control structures	• **2-AP-17** Systematically test and refine programs using a range of test cases **1B-AP-10** Create programs that include sequences, events, loops, and conditionals.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		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. 3. Develop problem solutions using a programming language, including all of the following: looping behavior, conditional statements, expressions, variables, and functions. **CSS.CT.3-5.5** Develop and employ strategies for understanding and solving problems in ways that use the power of technological methods to develop and test solutions.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		2. Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process. 4. Create programs that include sequences, events, loops, conditionals, and variables. **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. 3. Solve algorithmic problems of increasing complexity.	
Lesson 5: JavaScript Logic	• Learners will identify and modify IF THEN and IF THEN ELSE statements in JavaScript • Learners will use Comparison operators in JavaScript	**CSS.KC.6-8.20** Design, develop, debug and implement computer programs.	**2-AP-12** Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
Average Completion time: Approximately 3 hours	• Learners will use Boolean operators in JavaScript	1. Develop a working vocabulary of programming including flowcharting and/or storyboarding, coding, debugging, user interfaces, usability, variables, lists, loops, conditionals, programming language, events. 5. Implement a simple algorithm in a computer program. 6. Develop an event driven program. 7. Create a program that accepts user and/or sensor input and stores the result in a variable. 8. Create a computer program that implements a loop. 9. Develop a program that makes a decision based on data or user input. 10. Debug a program with an error.	**2-AP-14** Create procedures with parameters to organize code and make it easier to reuse. **2-AP-17** Systematically test and refine programs using a range of test cases **1B-AP-10** Create programs that include sequences, events, loops, and conditionals.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		CSS.CT.6-8.37 Use and compare simple coding control structures 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. 3. Develop problem solutions using a programming language, including all of the following: looping behavior, conditional statements, expressions, variables, and functions. **CSS.CT.3-5.5**	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		Develop and employ strategies for understanding and solving problems in ways that use the power of technological methods to develop and test solutions. 2. Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process. 4. Create programs that include sequences, events, loops, conditionals, and variables. **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. 3. Solve algorithmic problems of increasing complexity.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
Lesson 6: Errors Average Completion time: Approximately 3 hours	• Learners will define and Identify Semantic Errors • Learners will define and Identify Syntax Errors	**CSS.KC.6-8.20** Design, develop, debug and implement computer programs. 1. Develop a working vocabulary of programming including flowcharting and/or storyboarding, coding, debugging, user interfaces, usability, variables, lists, loops, conditionals, programming language, events. 10. Debug a program with an error. **CSS.CT.6-8.37** Use and compare simple coding control structures 2. Create a program individually and collaboratively using a text-based programming. language; Identify variables and compare the types of data stored as variables.	**2-AP-10** Use flowcharts and/or pseudocode to address complex problems as algorithms. **2-AP-17** Systematically test and refine programs using a range of test cases **2-AP-19** Document programs in order to make them easier to follow, test, and debug.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		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. 3. Develop problem solutions using a programming language, including all of the following: looping behavior, conditional statements, expressions, variables, and functions. **CSS.CT.3-5.5** Develop and employ strategies for understanding and solving problems in ways that use the power of technological methods to develop and test solutions. 2. Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		4. Create programs that include sequences, events, loops, conditionals, and variables. **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. 3. Solve algorithmic problems of increasing complexity.	
Lesson 7: Project Average Completion time: Approximately 5 - 10 hours	Learners will Design and develop a programmatic solution for a problem using JavaScript.	**CSS.KC.6-8.20** Design, develop, debug and implement computer programs. 1. Develop a working vocabulary of programming including flowcharting and/or storyboarding, coding, debugging, user interfaces, usability, variables, lists,	**2-AP-13** Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. **2-AP-12** Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		loops, conditionals, programming language, events. 5. Implement a simple algorithm in a computer program. 6. Develop an event driven program. 7. Create a program that accepts user and/or sensor input and stores the result in a variable. 8. Create a computer program that implements a loop. 9. Develop a program that makes a decision based on data or user input. 10. Debug a program with an error. **CSS.CT.6-8.37** Use and compare simple coding control structures 2. Create a program individually and collaboratively using a text-based	**2-AP-14** Create procedures with parameters to organize code and make it easier to reuse. **2-AP-17** Systematically test and refine programs using a range of test cases **1B-AP-10** Create programs that include sequences, events, loops, and conditionals.

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		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. 3. Develop problem solutions using a programming language, including all of the following: looping behavior, conditional statements, expressions, variables, and functions. **CSS.CT.3-5.5** Develop and employ strategies for understanding and solving problems in ways that use the power of technological methods to develop and test solutions. 2. Decompose (break down) problems into smaller, manageable subproblems to	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		facilitate the program development process. 4. Create programs that include sequences, events, loops, conditionals, and variables. **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. 3. Solve algorithmic problems of increasing complexity. **CSS.CT.6-8.47** Consider others' perspectives as well as one's own perspective when developing computational solutions. **CSS.CT.6-8.42** Improve teamwork and collaboration skills: providing useful feedback, integrating	

© Prodigy Learning | prodigylearning.com | codingcredentials.com

Lesson Title	Lesson Objectives	GA Computer Science Standards	CSTA Standards
		feedback, understanding, and accepting multiple perspectives. **CSS.KC.6-8.18** Recognize that there may be multiple approaches to solving a problem.	

© Prodigy Learning | prodigylearning.com | codingcredentials.com