Introduction to Computer Science in JavaScript Scope and Sequence

The CodeHS introduction to computer science curriculum teaches the foundations of computer science and basic programming, with an emphasis on helping students develop logical thinking and problem solving skills. Once students complete the CodeHS Introduction to Computer Science course, they will have learned material equivalent to a semester college introductory course in Computer Science and be able to program in JavaScript.

Students learn the fundamentals of programming with an emphasis on problem solving and logical thinking. Topics covered include: graphics, animation and games, data structures, and more.

Module 1: Programming with Karel	
35 hours (7 weeks)	Teaches what it means to "program" and allows students to focus on solving problems using code, rather than getting bogged down in syntax. Students solve problems by moving Karel the Dog around the grid.
CSTA Standards Addressed 2-AP-10 Use flowcharts and/or pseudocode to address complex problems as algorithms. 2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. 3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. 3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs. 3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. 3A-CS-01 Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.	

3B-AP-14 Construct solutions to problems using student-created components, such as procedures, modules and/or objects. 3B-AP-16 Demonstrate code reuse by creating programming solutions using libraries and APIs.

Module 2: Karel Challenges	
10 hours (2 weeks)	Solve large and more complex problems using Karel. Several Karel challenges to tie everything learned in the Karel module together.
CSTA Standards Addressed 3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. 3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. 3B-AP-14 Construct solutions to problems using student-created components, such as procedures, modules and/or objects. 3B-AP-16 Demonstrate code reuse by creating programming solutions using libraries and APIs.	

Module 3: Basic JavaScript and Graphics	
10 hours (2 weeks)	Introduces the basics of JavaScript, including variables, user input, control structures, functions with parameters and return values, and basic graphics, how to send messages to objects.
CSTA Standards Addressed 2-AP-11 Create clearly named variables that represent different data types and perform operations on their values. 2-AP-19 Document programs in order to make them easier to follow, test, and debug. 3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.	

3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs. 3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. 3B-AP-14 Construct solutions to problems using student-created components, such as procedures, modules and/or objects. 3B-AP-16 Demonstrate code reuse by creating programming solutions using libraries and APIs.

Module 4: JavaScript Control Structures	
20 hours (4 weeks)	Learn about booleans, for loops, conditionals, nested control structures, and while loops. Use comparison and logical operators to control the flow of the program.
CSTA Standards Addressed 2-AP-11 Create clearly named variables that represent different data types and perform operations on their values. 2-AP-14 Create procedures with parameters to organize code and make it easier to reuse. 2-AP-19 Document programs in order to make them easier to follow, test, and debug. 3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. 3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. 3B-AP-12 Compare and contrast fundamental data structures and their uses. 3B-AP-16 Demonstrate code reuse by creating programming solutions using libraries and APIs.	

Module 5: Functions and Parameters	
20 hours (4 weeks)	Learn functions with and without parameters, functions with and without

	return values, nested control structures, and local variables and scope. Using various kinds of functions such as functions with and without parameters and functions with and without return values.

CSTA Standards Addressed

2-AP-11 Create clearly named variables that represent different data types and perform operations on their values.

2-AP-14 Create procedures with parameters to organize code and make it easier to reuse.

2-AP-19 Document programs in order to make them easier to follow, test, and debug.

3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.

3B-AP-12 Compare and contrast fundamental data structures and their uses.

3B-AP-14 Construct solutions to problems using student-created components, such as procedures, modules and/or objects.

3B-AP-16 Demonstrate code reuse by creating programming solutions using libraries and APIs.

Module 6: JavaScript and Graphics Challenges	
5 hours (1 weeks)	Solve large and more complex problems using JavaScript and graphics. Graphics challenges tie everything in the module toegher.

CSTA Standards Addressed

2-AP-11 Create clearly named variables that represent different data types and perform operations on their values.

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-19 Document programs in order to make them easier to follow, test, and debug.

3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests. 3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. 3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. 3B-AP-12 Compare and contrast fundamental data structures and their uses. 3B-AP-14 Construct solutions to problems using student-created components, such as procedures, modules and/or objects. 3B-AP-16 Demonstrate code reuse by creating programming solutions using libraries and APIs.

Module 7: Animation and Games	
25 hours (5 weeks)	Learn timers, randomizing games, mouse events, and keyboard events. Use timers to add randomizations to graphical programs, use mouse events for interactive programs, use keyboard events for interactive programs.
CSTA Standards Addressed 2-AP-11 Create clearly named variables that represent different data types and perform operations on their values. 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-19 Document programs in order to make them easier to follow, test, and debug. 3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests. 3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. 3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.	

3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.

3B-AP-12 Compare and contrast fundamental data structures and their uses.

3B-AP-14 Construct solutions to problems using student-created components, such as procedures, modules and/or objects.

3B-AP-16 Demonstrate code reuse by creating programming solutions using libraries and APIs.

3B-AP-19 Develop programs for multiple computing platforms.

Module : Project - Breakout	
20 hours (4 weeks)	Learn timers, randomizing games, mouse events, and keyboard events. Use timers to add randomizations to graphical programs, use mouse events for interactive programs, use keyboard events for interactive programs.

CSTA Standards Addressed

2-AP-11 Create clearly named variables that represent different data types and perform operations on their values.

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-19 Document programs in order to make them easier to follow, test, and debug.

3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.

3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.

3B-AP-12 Compare and contrast fundamental data structures and their uses.

3B-AP-14 Construct solutions to problems using student-created components, such as procedures, modules and/or objects.

3B-AP-16 Demonstrate code reuse by creating programming solutions using libraries and APIs.

3B-AP-19 Develop programs for multiple computing platforms.