CodeHS

Intro to Computer Science with JavaScript Syllabus
Collie: One Semester, 75 hours

Course Overview and Goals

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.

Learning Environment: The course utilizes a blended classroom approach. The content is fully web-based, with students writing and running code in the browser. Teachers utilize tools and resources provided by CodeHS to leverage time in the classroom and give focused 1-on-1 attention to students. Each unit of the course is broken down into lessons. Lessons consist of video tutorials, short quizzes, example programs to explore, and written programming exercises. Each unit ends with a comprehensive unit test that assesses student's mastery of the material from that unit.

Programming Environment: Students write and run JavaScript programs in the browser using the CodeHS editor.

More information: Browse the content of this course at https://codehs.com/course/20519

Prerequisites

The Intro to Computer Science in JavaScript course is designed for complete beginners with no previous background in computer science. The course is highly visual, dynamic, and interactive, making it engaging for new coders.

Course Breakdown

Unit 1: Introduction to Programming in JavaScript with Karel the Dog (3 weeks/15 hours)

Students learn the basics of programming by giving Karel the Dog commands in a grid world.

Objectives / Topics Covered	• Commands • Defining vs. Calling Methods • Designing methods • Program entry points • Control flow • Looping • Conditionals • Commenting code • Preconditions and Postconditions • Top Down Design
Assignments / Labs	• 26 Karel programming exercises in total • Program-specific tasks for Karel the Dog ○ Example Exercise: Pyramid of Karel Write a program to have Karel build a pyramid. There should be

	three balls on the first row, two in the second row, and one in the third row. ● Teach Karel new commands like `turnRight()` or `makePancakes()` ○ Example Exercise: Pancakes Karel is the waiter. He needs to deliver a stack of pancakes to the guests on the 2nd, 4th, and 6th avenue. Each stack of pancakes should have three pancakes. Create a method called `makePancakes()` to help Karel solve this problem. ● Solve large Karel problems by breaking them down into smaller, more manageable problems using Top Down Design ○ Example Exercise: The Two Towers In this program, Karel should build two towers of tennis balls. Each tower should be 3 tennis balls high. At the end, Karel should end up on top of the second tower, facing East. ● Using control structures and conditionals to solve general problems ○ Example Exercise: Random Hurdles Write a program that has Karel run to the other side of first street, jumping over all of the hurdles. However, the hurdles can be in random locations. The world is fourteen avenues long.

Unit 2: Karel Challenges (1.5 weeks, 7 hours)
Students apply all the foundational concepts from Intro to Karel to solve new challenges.

Objectives / Topics Covered	● Solving large and more complex problems using Karel
Assignments / Labs	● 5 Karel challenges to tie everything learned in the Karel module together ○ Example Exercise: Super Cleanup Karel Karel's world is a complete mess. There are tennis balls all over the place, and you need to clean them up. Karel will start in the bottom left corner of the world facing east, and should clean up all of the tennis balls in the world. This program should be general enough to work on any size world with tennis balls in any locations.

Unit 3: JavaScript Basics (1 week/5 hours)
Students learn the basics of JavaScript including variables, user input, mathematics, and functions.

Objectives / Topics Covered	● Variables ● User Input ● Arithmetic Expressions ● Constants ● Collaborative Programming ● Random Numbers ● Functions
Assignments / Labs	● 12 JavaScript programming exercises in total ● Using variables and getting user input using JavaScript ○ Example Exercise: Dinner Plans Prompt the user for their name, then ask them what time you should meet for dinner.

	Greet them by name and tell them you will meet them at the time they specified!

Unit 4: The Canvas and Graphics (1 week/5 hours)
Students learn how to add graphics objects and position them on the canvas.

Objectives / Topics Covered	● JavaScript Canvas ● JavaScript Graphics ● Positioning Graphics Objects
Assignments / Labs	● 7 JavaScript programming and graphics exercises in total ○ Example Exercise: Create Your Own Meme In this exercise, you are going to create your own meme! The only requirements are that you add at least one image and one text element.

Unit 5: Graphics Challenges (1 week/5 hours)
Students apply what they have learned about graphics and basic JavaScript to complete a set of challenges.

Objectives / Topics Covered	● Solving large and more complex problems using graphics
Assignments / Labs	● 3 graphics challenges to tie everything learned in the JavaScript & Graphics module together ○ Example Exercise: Ghost Write a program to draw a ghost on the screen. You must do this by using the constant values given (this will allow us to easily alter the size or color of the ghost.)

Unit 6: JavaScript Control Structures (3 weeks/15 hours)
Students learn how to use control structures such as if/else statements and loops to make advanced programs in JavaScript.

Objectives / Topics Covered	● Booleans ● If/Else Statements ● Logical Operators ● Comparison Operators ● Conditionals ● While Loops ● Break Statements ● For Loops ● Nested Control Structures
Assignments / Labs	● 31 control structures programming exercises in total ● Using comparison and logical operators to control the flow of the program ○ Example Exercise: Inventory Write a program that keeps track of a simple inventory for a store. While there are still items left in the inventory, ask the user how many items they would like to buy. Then print out how many are left in inventory after the purchase. You should use a while loop for this problem.

	Make sure you catch the case where the user tries to buy more items than there are in the inventory. In that case, you should print a message to the user saying that their request isn't possible. ● Using for loops ○ Example Exercise: Jukebox ■ In the days before the internet, many restaurants would have a jukebox that allowed customers to choose what music they wanted to play. Customers would enter a coin and choose from the jukebox's music collection by selecting a song's number. You could choose one song per coin. In this exercise, you will create a digital jukebox where the user can enter any number of quarters to create a playlist of songs. ● Drawing basic graphics using JavaScript ○ Example Exercise: Caterpillar This graphics program should draw a caterpillar. A caterpillar has NUM_CIRCLES circles. Every other circle is a different color, the even circles are red, and the odd circles are green (by even we mean when i is an even number). Use a for loop to draw the caterpillar, centered vertically on the screen. Also, be sure that the caterpillar is still drawn across the whole canvas even if the value of NUM_CIRCLES is changed.

Unit 7: Control Structures Challenges (1 week/5 hours)
Students apply the foundational concepts from the Control Structures module to solve new challenges.

Objectives / Topics Covered	● Solving large and more complex problems using control structures
Assignments / Labs	● 3 challenges using control structures to tie everything learned in the JavaScript Control Structures module together ○ Example Exercise: Guessing Game The computer picks a number between 1 and 100, and you have to guess it. The computer will tell you whether your guess was too high, too low, or correct. Your assignment is to generate a random number and let the user guess numbers until they guess the correct number. Make sure to let the user know what they should do at the beginning of the program!

Unit 8: Functions (2 weeks/10 hours)
Students learn to write reusable code with functions, parameters, and return values, and explore the impact of variable scopes.

Objectives / Topics Covered	● Parameters ● Return Values ● Default Parameters ● Scope
Assignments / Labs	● 12 functions programming exercises in total ● Using various kinds of functions such as functions with and without parameters, and functions with and without return values ○ Example Exercise: Is it even? Write a program that continually asks the user for integers and then

	prints whether their input is even or odd. The user should keep entering numbers until they enter 0; at that point, print "Done!" on its own line. In order to check if the inputted integer is even or odd, you should define and call a function named \`isEven()\`. This function should return a Boolean value of \`true\` or \`false\` depending if the number is even or not.

Unit 9: Functions Challenges (1 week/5 hours)
Students use what they have learned in the Functions module to solve new challenges.

Objectives / Topics Covered	● Solving large and more complex problems using functions
Assignments / Labs	● 3 challenges using functions to tie everything learned in the Functions module together ○ Example Exercise: Balloons You should use lines, circles, and random colors to draw a bunch of balloons. All the balloon strings should start two-thirds down the canvas. Each string line should travel upward to a random point and have a circle placed on top of the endpoint. Each balloon should be a random color and have a radius between \`MIN_RADIUS\` and \`MAX_RADIUS\`.

Unit 10: FInal Project (1-2 weeks, 5-10 hours)
Students apply what they have learned in JavaScript to program graphics and game challenges.

Objectives / Topics Covered	● Review of JS graphics ● Review of functions ● Review of top-down design
Assignments / Labs	● Final Project: Students will use what they learned in the course to brainstorm, plan, and implement a JS graphics program that draws a picture. Students will go through a brainstorming process, and then select their final idea. They will establish milestones in the completion of their project, pseudocode their project before they finally implement the code to complete their goal.

Sample of Optional Supplemental Materials (Remainder of school year)

Supplemental Modules	● Extra practice with: ○ Karel ○ Basic JavaScript ○ Control Structures ○ Functions ○ Graphics ● Animation and Games ● Data Structures: Arrays ● Data Structures: Objects ● Visualizing Music