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. Once students complete the 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.

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 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, adding up to over 100 hours of hands-on programming practice in total. 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 editor.

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 (8 weeks/40 hours

Objectives / Topics Covered	● Commands ● Defining vs. Calling Methods ● Designing methods ● Program entry points ● Control flow ● Looping ● Conditionals ● Classes ● Commenting code ● Preconditions and Postconditions ● Top Down Design
Assignments / Labs	● 30 Programming Exercises and Challenges in total ● 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. ○ 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 2: Basic Javascript and Graphics (12 weeks/60 hours)

Objectives / Topics Covered	● Variables ● User Input ● Arithmetic Expressions ● Drawing basic graphics ● Booleans ● Random Numbers ● Loops ● Conditionals ● Functions with and without parameters ● Functions with and without return values ● Nested Control Structures ● Local variables and scope
Assignments / Labs	● 1-3 exercises per topic for a total of 41 exercises. ● Using variables and getting user input using JavaScript ○ Example Exercise: Grocery Store Prompt the user for their name, and then how many apples, and then how many oranges they would like to buy. Then print out the name that was given, as well as how many apples and oranges they wanted. ● Drawing basic graphics using JavaScript ○ Example Exercise: French Flag This program should draw the French flag. The left third of the canvas is blue, the middle third is white, and the right third is red. You will need to use Rectangle objects in this program. ○ 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 in the screen. Also, be sure that the caterpillar is still drawn across the whole canvas even if the value of NUM_CIRCLES is changed. ● 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 various kinds of functions such as functions with and without parameters, and functions with and without return values
    - Example Exercise: Horizontal Lines
      Write a function that draws horizontal lines on the graphics canvas. If a line is horizontal, then the y-values for the endpoints are the same.
      The parameters to your function should be the y location, and the length, and all of your lines should start at x position 0.
    - Example Exercise: Is it even?
      Write a function called isEven that returns a boolean of whether or not a value is even or odd. The isEven function should not print anything out or return a number. It should only take in a number and return a boolean.
      Once you've written this function, write a program that asks the user for integers and prints whether the number they entered is even or odd using your isEven function. You should let the user keep entering numbers until they enter the SENTINEL given.
- Graphics Challenges to tie everything in the module together
    - Example Exercise: Ghosts
      Write a program to draw ghosts on the screen. You must do this by writing a function called drawGhost, which takes three parameters, the center x location of the ghost, the center y location of the ghost and the color of the ghost.

Unit 3: Animation and Games (5 weeks/25 hours)

Objectives / Topics Covered	● Timers ● Randomizing Games ● Mouse Events ● Keyboard Events
Assignments / Labs	● 13 exercises in total ● Using timers to add randomizations to graphical programs ○ Example Exercise: Paint Splatter Write a program that splatters paint on the screen every DELAY milliseconds. To splatter paint, pick a random color and draw CIRCLES_PER_SPLATTER circles of that color at random places on the screen. The radius of each circle should be a random value between MIN_RADIUS and MAX_RADIUS. Remember to use helper functions. ● Using mouse events for interactive programs ○ Example Exercise: Teleporting Ball Extend our bouncing ball program. Whenever you click, the ball should teleport to that spot and change to a random color. ○ Example Exercise: Target Draw a target on the screen that moves to aim at where your mouse is located. A target consists of a horizontal line that goes from 0 to the window width and a vertical line that goes from 0 to the window height. The lines should cross paths where the mouse is. If you're feeling adventurous, you can extend this to draw a small red circle whenever you click. If you're feeling really adventurous, you can have a bouncing ball on the screen and see if you can remove it when it gets clicked. You can use remove(obj) to remove something from the screen and getElementAt(x, y) to get an object at the given position. It will return the object or will return null if there is no object there. ● Using keyboard events for interactive programs ○ Example Exercise: Basic Snake Write a basic version of the snake game. The way our game works is by first creating a green square at the center of the screen. The snake should be moving to the right. If you hit an arrow key, you should change the snake's direction.

Unit 4: Project: Breakout (4 weeks/20 hours)

Objectives / Topics Covered	● Basic graphics ● Mouse events ● Collision detection
Assignments / Labs	● Guided exercises to build a Breakout Game ● The Breakout Game is made up of bricks at the top of the screen, a paddle that you control at the bottom of the screen, and a ball that bounces around. Your goal is to direct the paddle with your mouse to bounce the ball until all of the bricks have been hit and disappear.

Unit 5: Basic Data Structures (6 weeks/30 hours)

Objectives / Topics Covered	● List/Array creation and basic operations ● Iterating through lists/arrays ● Finding and removing elements in lists/arrays ● Object/Map basics ● Iterating over an object/map ● Set creation and basic operations ● Grid creation and basic operations
Assignments / Labs	● 24 exercises in total ● Basic list/array operations ○ Example Exercise: List of Places to Travel Create an array of the top 5 places you would like to travel called travelList. Print out the item at index 2. ● Iterating through arrays/lists ○ Example Exercise: Draw a Barcode In this program, you will draw a barcode on the screen given an array that represents the data in the barcode. The array will contain a boolean in it, and if the boolean is true, you will draw a vertical line in that position that runs from the top to the bottom of the screen. If not, you will not draw a line. We have written the generateBarcode function for you that creates a random barcode. Your job is to write the drawBarcode function. ● Basic list/array algorithms ○ Example Exercise: Remove From Line ou are given an array of names of people who are in line for movie tickets. Use the remove element to remove the first person from the line twice, as if you

	have just given them their tickets. You should write a function to print everyone in line. Then print the line before and after removing the people. ● Basics of objects/maps ○ Example Exercise: Sidekicks Police Commissioner Gordon has tasked you with building up a database of superhero sidekicks, just in case the superheros are all busy. Given a superhero name, we need to be able to look up the name of that superhero’s sidekick. We’ve already started the database for you, but you need to add to it. Add an entry to the sidekicks Object for Batman’s sidekick Robin. Get the name of Mermaid Man’s sidekick and print it out Who is your sidekick? Add their name to the sidekicks Object as your sidekick ● Basics of sets ○ Example Exercise: Mutual Friends Write a program that prints the mutual friends between two people. You should create two sets, one for each person, and add friends (strings) to each set representing the friends of that person. Then, fill in the mutualFriends function that takes the two people as parameters and returns a new set that includes their mutual friends. Print out the set of mutual friends ● Iterating through a Grid ○ Example Exercise: Summing Grid Write a function called function sumGrid(grid) that takes a grid as a parameter and fills each location in the grid with the sum of the row index and column index of that location.

Unit 5: Project - Tic Tac Toe (2 weeks/10 hours)

Objectives / Topics Covered	● Using data structures to solve a problem ● Combining data structures and graphics
Assignments / Labs	● Guided exercises to build a game of Tic Tac Toe

Unit 6: Game Design Components Project: Helicopter (3 weeks/15 hours)

Objectives / Topics Covered	● Basic Graphics ● Collision detection ● Scrolling background ● Generating random obstacles
Assignments / Labs	● Guided exercises to explain the basic elements of game design and build a Helicopter Game. ● Helicopter Game is played by controlling a helicopter with the mouse to navigate through a changing terrain and flying obstacles.