

Jumping for JavaScript

Web Development Boot Camp
Lesson 3.2



Today's Class

Objectives

In today's class, we'll cover:



Array Assignments



The Concept of for Loops



The Art of Pseudocoding



Building Rock-Paper-Scissors

Basics Recap



What is JavaScript?

(And what is it used for?)

JavaScript Definitions



JavaScript is the third of the three fundamental programming languages of the modern web (along with HTML and CSS).



JavaScript allows developers to create dynamic web applications capable of taking in user inputs, changing what's displayed to users, animating elements, and much more.



What is a Variable?

(And how do we declare one?)

Variable Basics



Variables are the nouns of programming.



They are “things” (numbers, strings, Booleans, etc.).



A variable is composed of a variable name and a value.

```
var name = "Snow White";  
var dwarfCount = 7;  
var isSleeping = true;
```


Variable Basics: Syntax

Var keyword	Variable name	Assignment	Value	Termination
<i>var</i>	name	=	"Snow White"	;



What is meant by `console.log`?
(And how does it differ from an alert, prompt, or confirm?)

Console.log

`console.log` is a quick expression that prints content to the debugger—very useful during development and debugging!

```
var quick = "Fox";  
var slow = "Turtle";  
var numbers = 121;  
  
// The console.log() method is used to display data in the the browser's console.  
// We can log strings, variables, and even equations.  
  
console.log("Teacher");  
console.log(quick);  
console.log(slow);  
console.log(numbers + 15);
```

Basic Variables



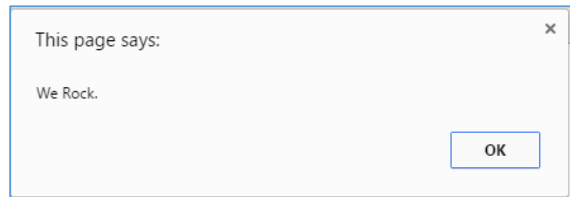
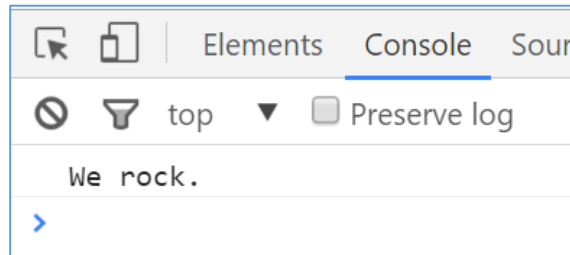
`console.log` displays discreetly to the debugger.



`alert` displays a pop-up message to the user.

```
console.log("We rock.");
```

```
alert("We Rock.");
```



Basic Variables



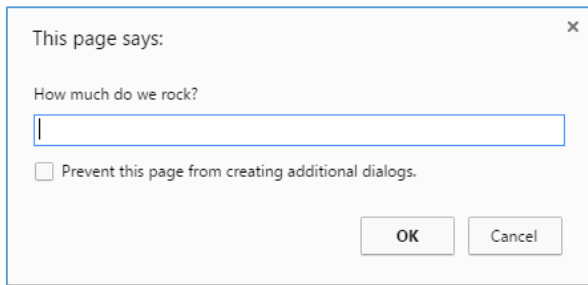
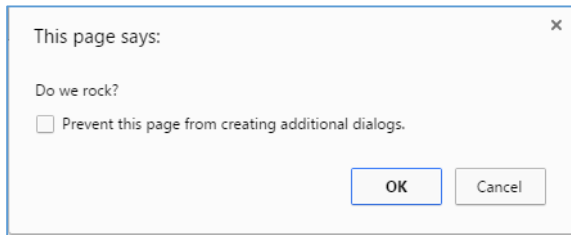
`confirm` displays a true/false popup.



`alert` displays a `prompt` with a text-box input.

```
confirm("Do we rock?");
```

```
prompt("How much do we rock?");
```





What is an **array**?

Basic Arrays



Arrays are a type of variable that are collections.



These collections can be made up of strings, numbers, Booleans, other arrays, objects, anything.



Each element of the array is marked by an **index**. Indexes always start with 0.

```
var nickCharacters = ["Tommy", "Doug", "Oblina"];
```

```
var diceNumbers = [1, 2, 3, 4, 5, 6,];
```

```
var mixedArray = ["Zoo", 12, "Carrot", 3];
```

Arrays: Indices



To recover the value at any specific index, include the name of the array with a square bracket `[]` and inside the bracket is the element's index.



You can easily grab the number of elements in the array using the method `array.length`.

```
// Our array of zoo animals.  
var zooAnimals = ["Zebra", "Rhino", "Giraffe", "Owl"];  
  
// Prints 4 to the console because there are 4 items in our zooAnimals array.  
console.log(zooAnimals.length);  
  
// Prints Rhino to the console. Remember, the first item in an array has an index position of 0!  
console.log(zooAnimals[1]);  
  
// Prints undefined...because the last index ("Owl") is 3.  
console.log(zooAnimals[4]);
```




How do we “**write**” text to the HTML itself?

Writing to HTML

We can also use JavaScript to directly write to the HTML page itself using `document.write()`. Later we will go over much more advanced approaches for writing HTML using JavaScript and jQuery.

Test.html (sublime)

```
<!DOCTYPE html>
<html lang="en-us">
  <head>
    <meta charset="UTF-8">
    <title>Document Write</title>
  </head>
  <body>

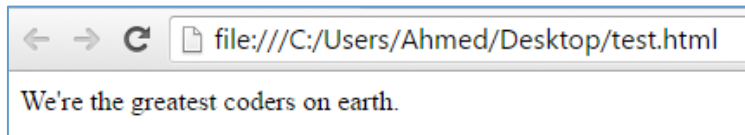
    <script type="text/javascript">

      document.write("We're the greatest coders on earth.");

    </script>

  </body>
</html>
```

Test.html (chrome)





How do we **check** conditions?

If/Else statements are critical

Each statement is composed of an if, else-if, or else (keyword), a condition, and the resulting code in { } curly brackets.

```
// If the user likes sushi (confirmSushi === true), we run the following block of code.
if (confirmSushi) {
    alert("You like " + sushiType + "!");
}
// If the user likes ginger tea (confirmGingerTea === true), we run the following block of code.
else if (confirmGingerTea) {
    alert("You like ginger tea!!");
}
// If neither of the previous condition were true, we run the following block of code.
else {
    document.write("You don't like sushi or ginger tea.");
}
```



Activity: Basic JavaScript Dissection

Suggested Time:
3 minutes



Activity: Basic JavaScript Dissection



Re-examine the file sent to you during yesterday's class.



See if you can better understand how it works—at this point in today's class.



Prepare to share once the time is up.

Suggested Time: 3 minutes





Activity: Array Logging

Suggested Time:
5 minutes



Activity: Array Logging



Follow the instructions provided in the file to `console.log` each of the names in the `coolPeople` variable.



Hint: You should be repeating the same line 6 times.



Be prepared to share once time is up.

Suggested Time: 5 minutes





Activity: Array Setting

Suggested Time:
7 minutes



Activity: Array Setting



Follow the instructions in the file provided to convert each item in the array to lowercase.



Make sure to only add in lines of code where instructed.



Hint: You will need to use the method `.toLowerCase()`. Research if you don't remember how to use it.



Be prepared to share when time is up.

Suggested Time: 7 minutes



Back to The Zoo Pen

Array Name: zooAnimals

Zebra

Index 0

Rhino

Index 1

Giraffe

Index 2

Owl

Index 3

Coded in JavaScript using an Array

```
// Our array of zoo animals.  
var zooAnimals = ["Zebra", "Rhino", "Giraffe", "Owl"];
```

Back to The Zoo Pen

Array Name: zooAnimals

Zebra

Index 0

Rhino

Index 1

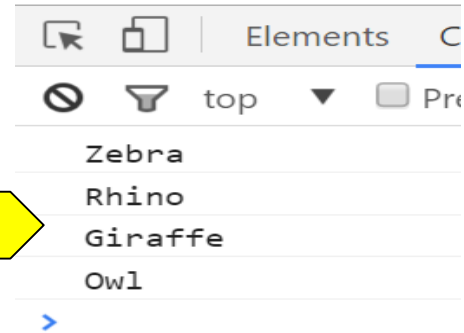
Giraffe

Index 2

Owl

Index 3

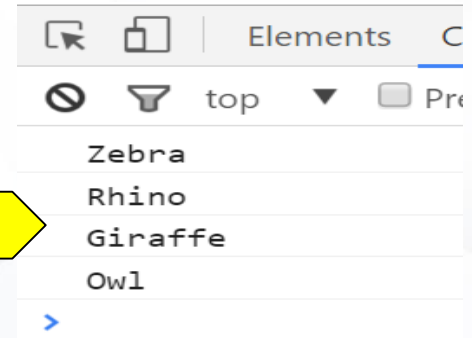
```
// Array of zoo animals.  
var zooAnimals = ["Zebra", "Rhino", "Giraffe", "Owl"];  
  
console.log(zooAnimals[0];  
console.log(zooAnimals[1];  
console.log(zooAnimals[2];  
console.log(zooAnimals[3];
```





What's wrong here?

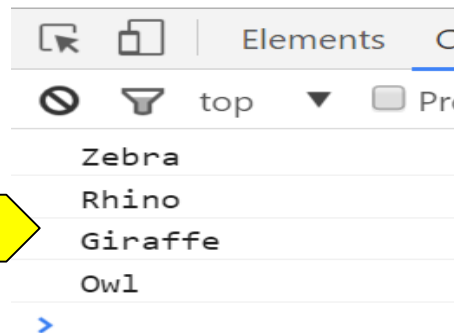
```
// Array of zoo animals.  
var zooAnimals = ["Zebra", "Rhino", "Giraffe", "Owl"];  
  
console.log(zooAnimals[0];  
console.log(zooAnimals[1];  
console.log(zooAnimals[2];  
console.log(zooAnimals[3];
```

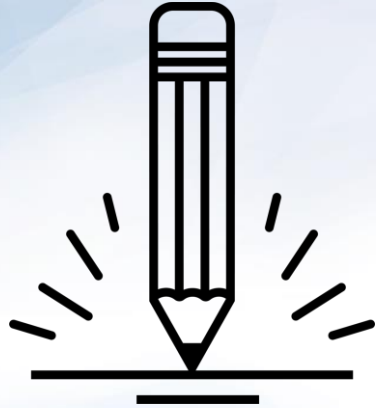


Don't Repeat Yourself (DRY)

Repeated Code! Let's be more efficient.

```
// Array of zoo animals.  
var zooAnimals = ["Zebra", "Rhino", "Giraffe", "Owl"];  
  
console.log(zooAnimals[0];  
console.log(zooAnimals[1];  
console.log(zooAnimals[2];  
console.log(zooAnimals[3];
```





Activity: For Loop Dissection

Suggested Time:
5 minutes



Activity: For Loop Dissection



Spend a few moments trying to dissect the code sent to you.



Think about what is happening with each line of code.



Feel free to do research if you are stumped. As a **hint**, look into the phrase “for loop”.



Be prepared to share when time is up.

Suggested Time: 5 minutes



Enter the For Loop

For loops are **critical** in programming.

We use for loops to run **repeated blocks of code** over a set period.

Each for loop is composed of a:



Variable declaration or counter
(iterator)



Loop condition



Iteration (addition)

```
// Start with an Array.  
var vegetables = ["Carrots", "Peas", "Lettuce", "Tomatoes"];  
  
// Loops through each index of the Array.  
for (var i = 0; i < vegetables.length; i++) {  
    console.log("I love " + vegetables[i]);  
}
```

Enter the For Loop

```
// Start with an Array.  
var vegetables = ["Carrots", "Peas", "Lettuce", "Tomatoes"];  
  
// Loops through each index of the Array.  
for (var i = 0; i < vegetables.length; i++) {  
  console.log("I love " + vegetables[i]);  
}  
  
// Logs:  
// I love Carrots  
// I love Peas  
// I love Lettuce  
// I love Tomatoes
```



Iterator

Condition

Increment

Enter the For Loop

Code between the `{ }` gets repeated each time the iterator is smaller than the condition (in this case, as long as `i < 4`).

```
// Start with an Array.  
var vegetables = ["Carrots", "Peas", "Lettuce", "Tomatoes"];  
  
// Loops through each index of the Array.  
for (var i = 0; i < vegetables.length; i++) {  
  console.log("I love " + vegetables[i]);  
}  
  
// Logs:  
// I love Carrots  
// I love Peas  
// I love Lettuce  
// I love Tomatoes
```

Enter the For-Loop

Running the code “loops” through and prints each element in the array.

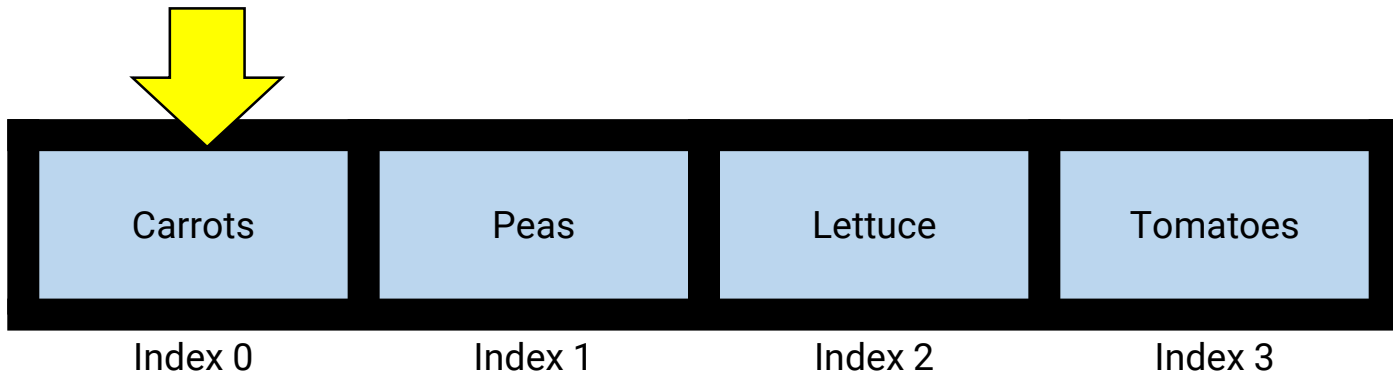
```
// Start with an Array.  
var vegetables = ["Carrots", "Peas", "Lettuce", "Tomatoes"];  
  
// Loops through each index of the Array.  
for (var i = 0; i < vegetables.length; i++) {  
    console.log("I love " + vegetables[i]);  
}
```

```
// Logs:  
// I love Carrots  
// I love Peas  
// I love Lettuce  
// I love Tomatoes
```

Run That Loop

```
// Start with an Array.  
var vegetables = ["Carrots", "Peas", "Lettuce", "Tomatoes"];  
  
// Loops through each index of the Array.  
for (var i = 0; i < vegetables.length; i++) {  
  console.log("I love " + vegetables[i]);  
}
```

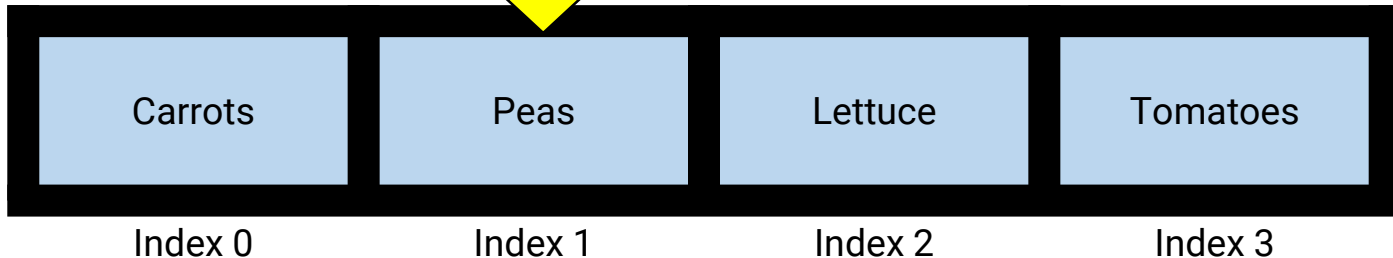
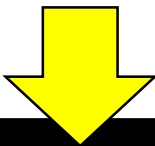
When $i = 0$... `console.log("I love Carrots")`



Run That Loop

```
// Start with an Array.  
var vegetables = ["Carrots", "Peas", "Lettuce", "Tomatoes"];  
  
// Loops through each index of the Array.  
for (var i = 0; i < vegetables.length; i++) {  
  console.log("I love " + vegetables[i]);  
}
```

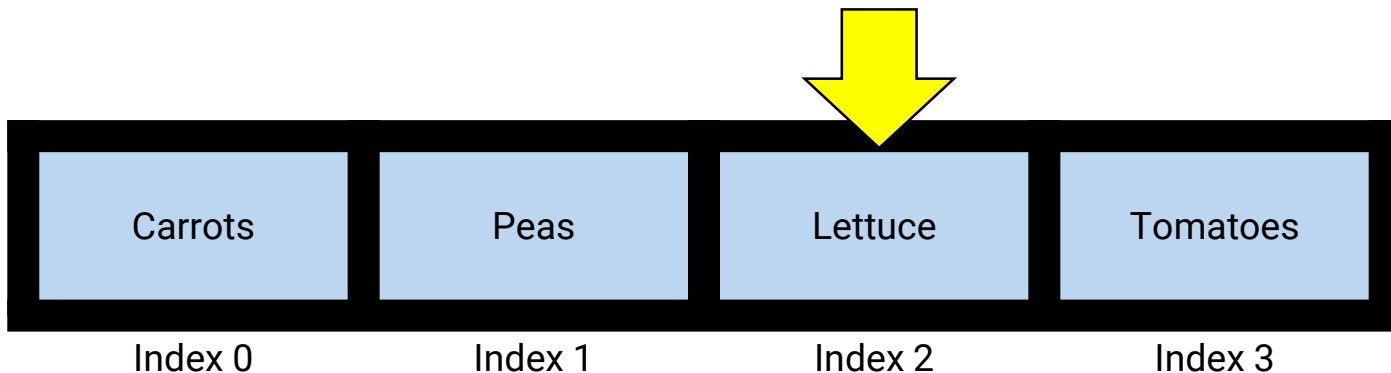
When $i = 1$... `console.log("I love Peas")`



Run That Loop

```
// Start with an Array.  
var vegetables = ["Carrots", "Peas", "Lettuce", "Tomatoes"];  
  
// Loops through each index of the Array.  
for (var i = 0; i < vegetables.length; i++) {  
  console.log("I love " + vegetables[i]);  
}
```

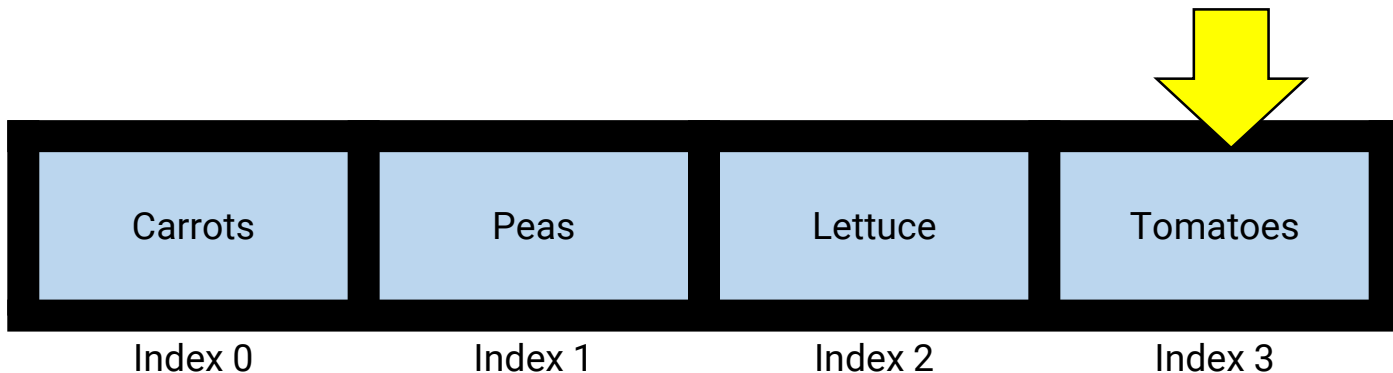
When $i = 2$... `console.log("I love Lettuce")`



Run That Loop

```
// Start with an Array.  
var vegetables = ["Carrots", "Peas", "Lettuce", "Tomatoes"];  
  
// Loops through each index of the Array.  
for (var i = 0; i < vegetables.length; i++) {  
  console.log("I love " + vegetables[i]);  
}
```

When $i = 3$... `console.log("I love Tomatoes")`





Activity: For Loop Zoo

Suggested Time:
15 minutes



Activity: For Loop Zoo

01

Spend a few moments rewriting the code below using a for loop.

02

If you need help, use the code from the previous example as a guide.

03

Then try to explain to the person next to you how your code works.

```
// Array of zoo animals.  
var zooAnimals = ["Zebra", "Rhino", "Giraffe", "Owl"];  
  
console.log(zooAnimals[0];  
console.log(zooAnimals[1];  
console.log(zooAnimals[2];  
console.log(zooAnimals[3];
```

Suggested Time: 15 minutes





Activity: Another Loop

Suggested Time:
15 minutes



Activity: Another Loop

Starting from scratch, create a for loop that prints the following lines:

I am 0

I am 1

I am 2

I am 3

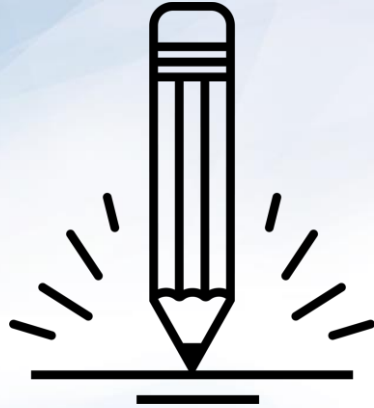
I am 4



This time, don't use an array!

Suggested Time: 15 minutes





Group Activity (2 people): Loop with Conditions

Suggested Time:
10 minutes



Group Activity: Loop with Conditions



Starting from scratch, write code that loops through the following array:

```
// This is our starting myFarm array.  
var myFarm = ["chickens", "pigs", "cows", "horses", "ostriches"];
```



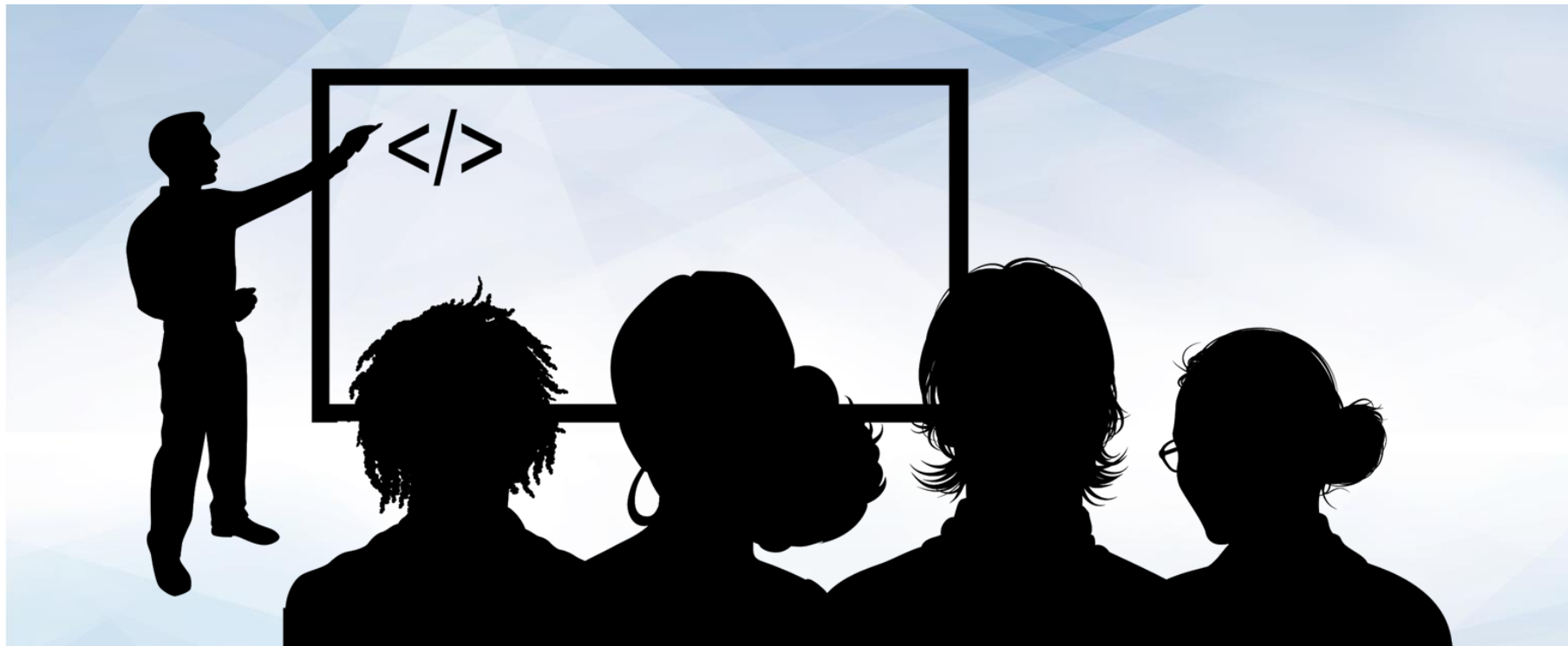
Use `console.log` to display the name of each animal on the farm.



Using the `.charAt()` method (research it), check if the first letter in the animal's name begins with a "c" or "o". If it does, create an alert saying: "Starts with c or an o!"

Suggested Time: 10 minutes





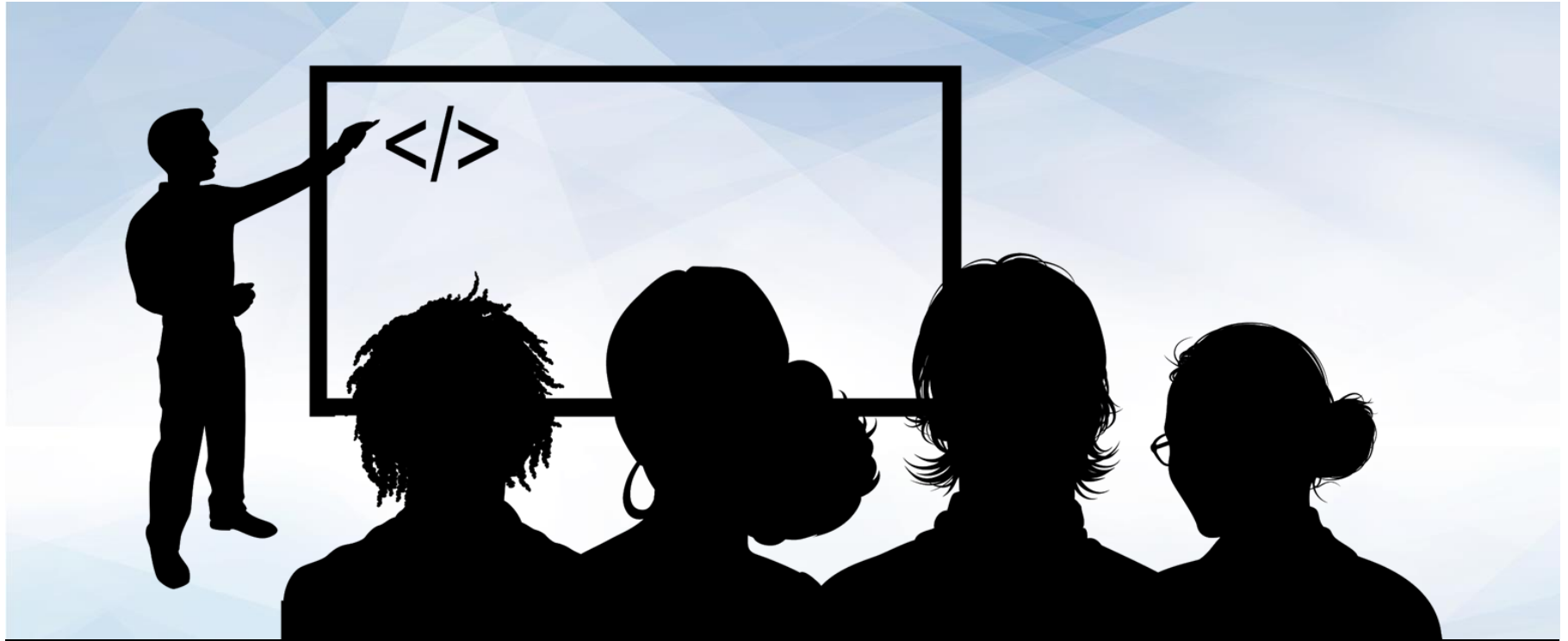
Instructor Demonstration

Events & DOM Manipulation

Rock-Paper-Scissors with a Partner

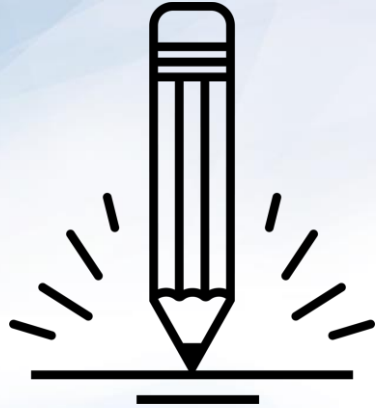
Play five rounds.





Instructor Demonstration

RPS Coded



Activity: Pseudocode Rock-Paper-Scissors (RPS)

Suggested Time:
10 minutes



Activity: Pseudocode RPS



Spend a few moments outlining all the steps and conditions that go into a single game of rock paper scissors.



Try to break it down into steps that you could “code out.”



Think of basic elements like loops, if-then statements, arrays, alerts, etc.



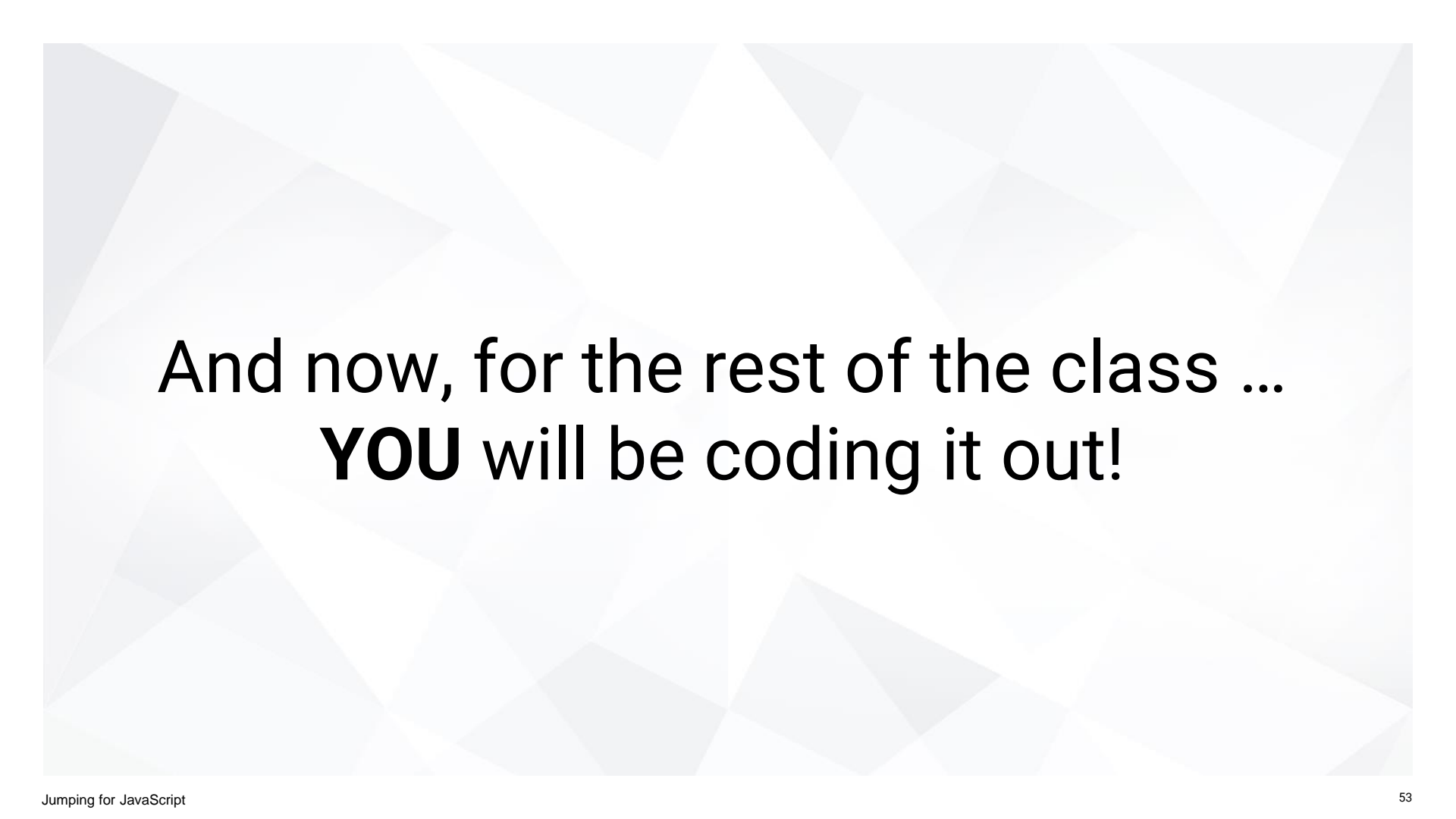
Be prepared to share your outlined approach.

Suggested Time: 8 minutes

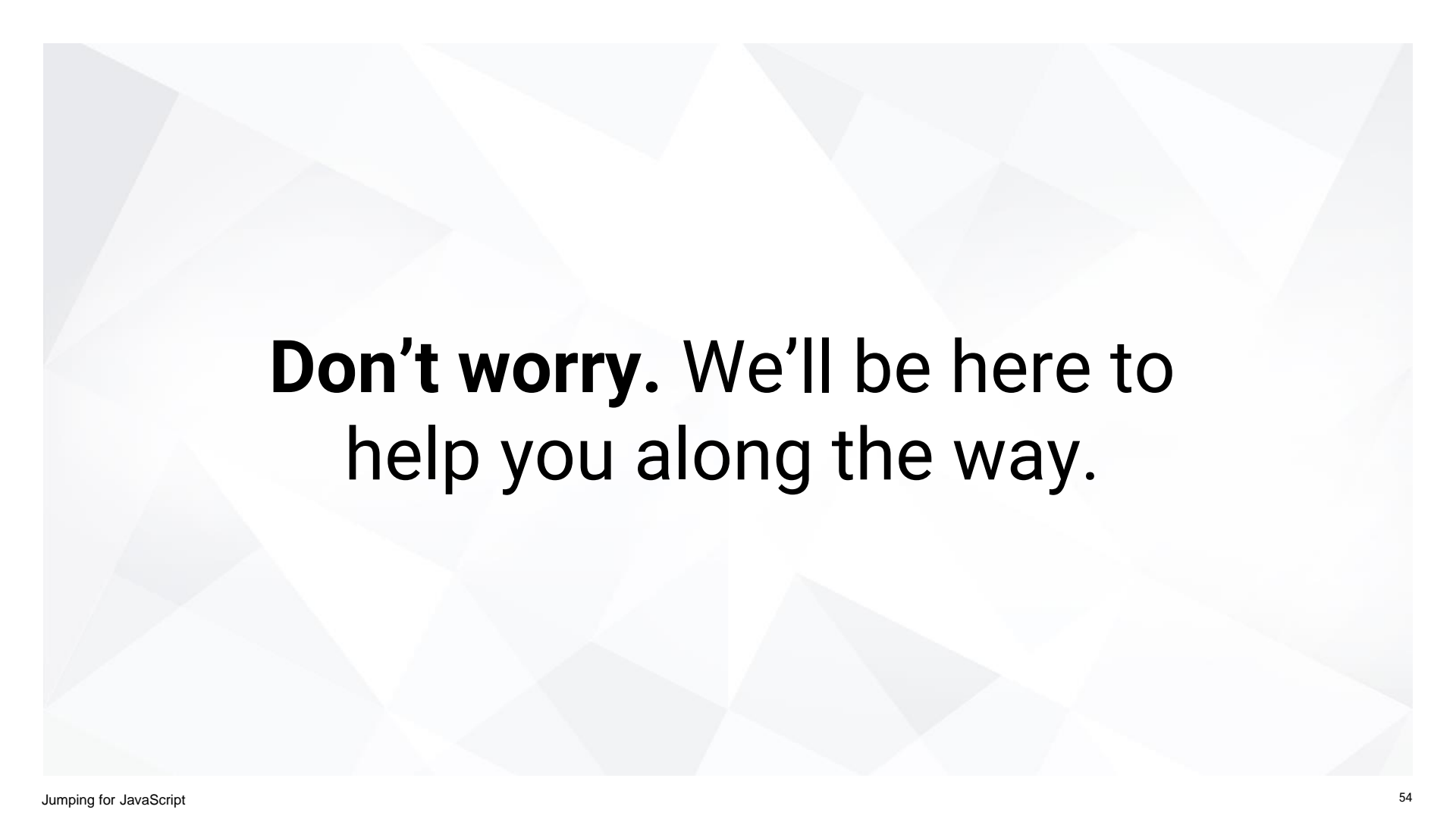


You just **pseudocoded!**






And now, for the rest of the class ...
YOU will be coding it out!



Don't worry. We'll be here to
help you along the way.

A black silhouette of a person standing on the peak of a jagged mountain, holding a flag aloft. A dashed white line representing a path leads up the mountain. The background is a light blue geometric pattern.

Group Challenge: Coding Out RPS

Suggested Time:
60 minutes



Group Challenge: Coding out RPS



In breakout groups, begin the process of coding out the rock-paper-scissors game.



Do as much as you can on your own, but don't be afraid to ask for help if you feel your team is struggling.



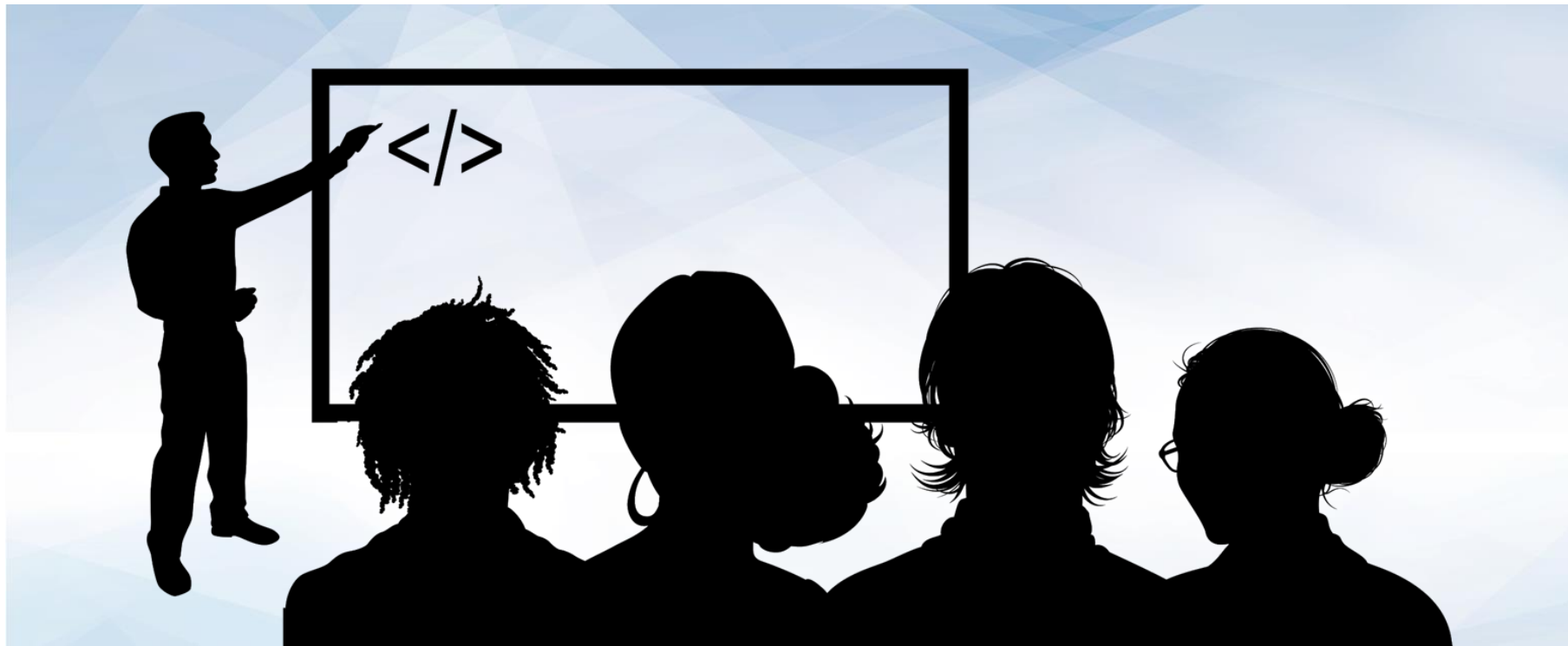
Note: Don't use `document.write` because it will delete the contents of your page, including your Javascript. Use `document.querySelector` or `document.getElementById` alongside either `innerHTML` or `textContent` to write to the DOM.



Don't worry. We know this is very challenging. We also know that you won't know where to start. In fact, we haven't shown you everything you need yet. But that's OK. Accepting confusion is a HUGE first step to becoming a coder!

Suggested Time: 60 minutes





Instructor Demonstration

Let's Fill in the Missing Code (Together)



Questions?