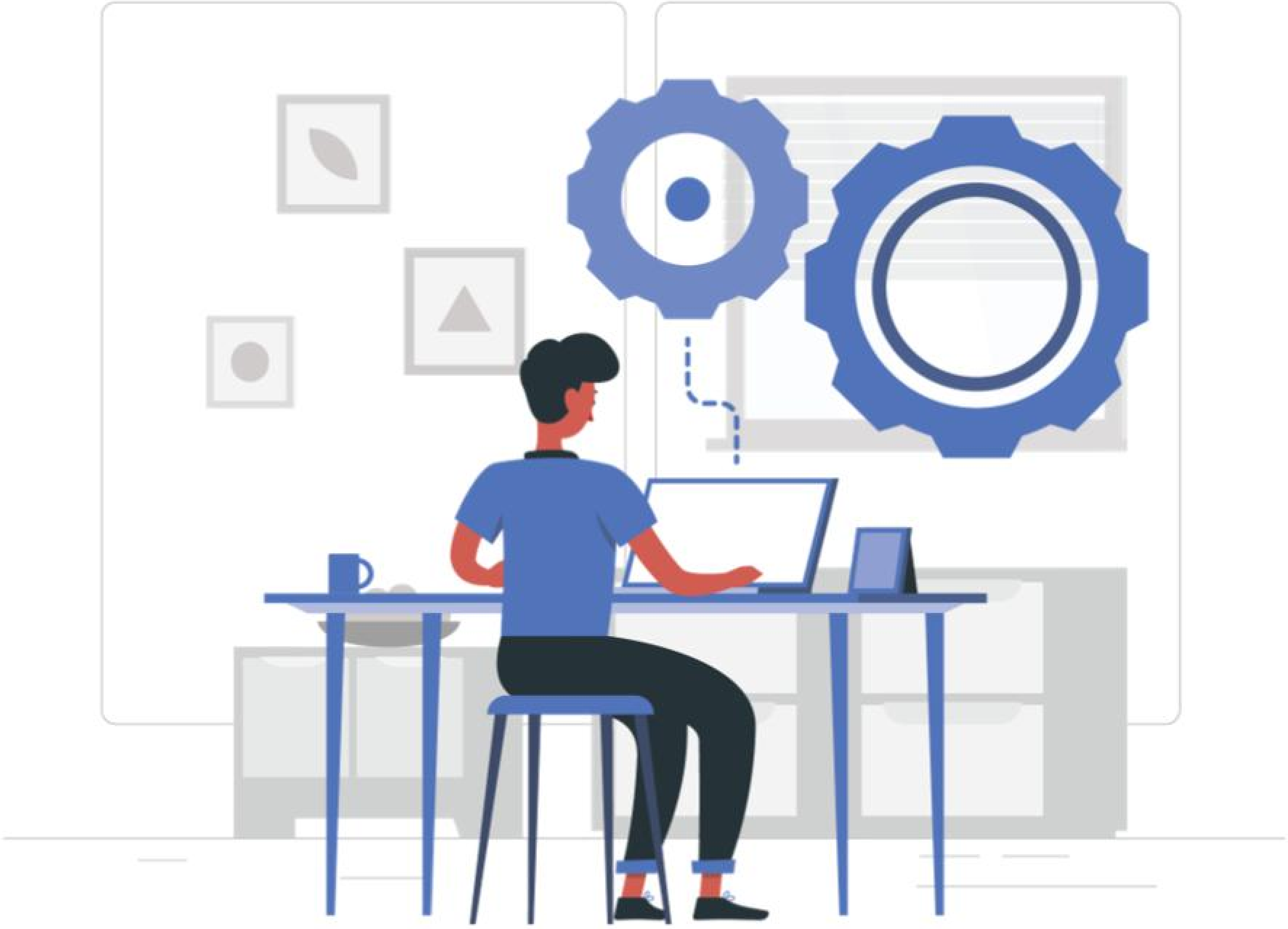


Practice

Utilize Arrays to Model Aggregate Data

Practices

- Create an object
- Calculate the average marks of the class
- Calculate the number of students whose marks >= 40
- Sort the given marks in the ascending order

Points to Remember

- Declare an array using an array literal.
- Declare an array using the `const` keyword and not the `let` keyword.
- Declare `for` loop variables within the loop statement and not outside the loop.
- Use the JavaScript debugging panel in DevTools to debug the code and rectify the error.
- Ensure the code is well indented and well commented and follows the JavaScript naming conventions for naming variables and constants.

PRACTICE

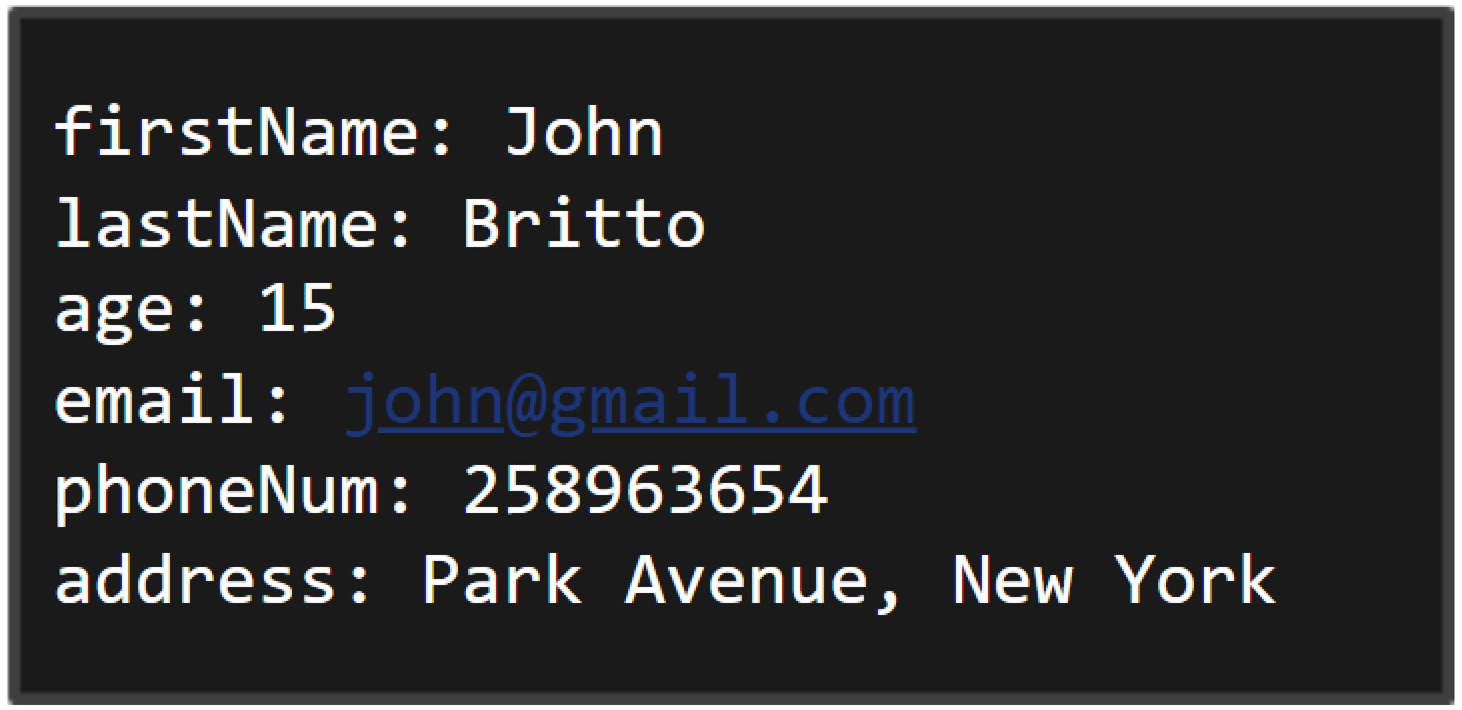
Practice 1: Create an Object

The administration of Grey River University wants to keep a record of all the students who have joined in their freshman year. To track the record of all their students' basic information, the University wants to develop a program.

Write a JavaScript program to create an object called student that has the following properties: firstName, lastName, age, email, phoneNum, and address.

Practice 1

- The solution for this practice should be written inside **p1-submission.js** located inside **p1-create-object** folder.
- Declare and initialize an object called student with the following properties: firstName, lastName, age, email, phoneNum, and address.
- Display each property value in the format given below:



```
firstName: John
lastName: Britto
age: 15
email: john@gmail.com
phoneNum: 258963654
address: Park Avenue, New York
```

PRACTICE

Practice 2: Calculate the Average Marks of the Class

A professor at Grey River University wants to calculate the average marks scored in geography by the students in their freshman year.

You need to write a JavaScript program to calculate the average marks of the class.

Note: Steps to do this practice are given in the upcoming slide.

Practice 2: Tasks

- The solution for this practice should be written inside **p2-submission.js** located inside **p2-average-marks** folder.
- Steps to perform this task:

1. Declare and initialize an array of size 10 with some random marks ranging between 0 and 100.
2. Initialize a sum variable with value 0.
3. Using a *for* loop, traverse the array and calculate the sum with each value traversed.
4. Calculate the average marks.

   averageMarks = sum / 10;

5. Return the average marks.

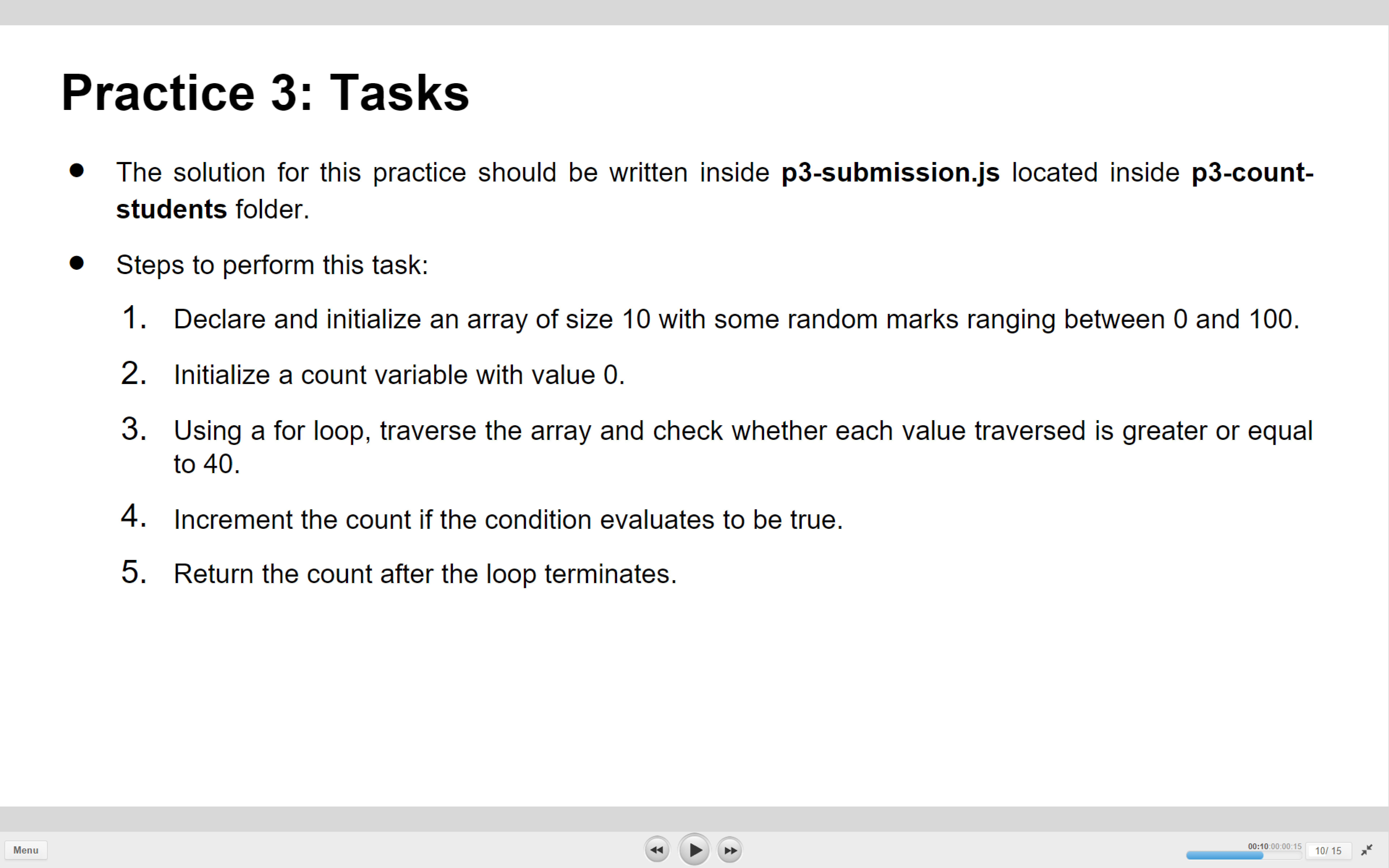
PRACTICE

Practice 3: Count of Students With Marks >= 40

A team of recruiters from an MNC has conducted a written test for college placements at Grey River University. They want to calculate the number of students who got Marks >= 40.

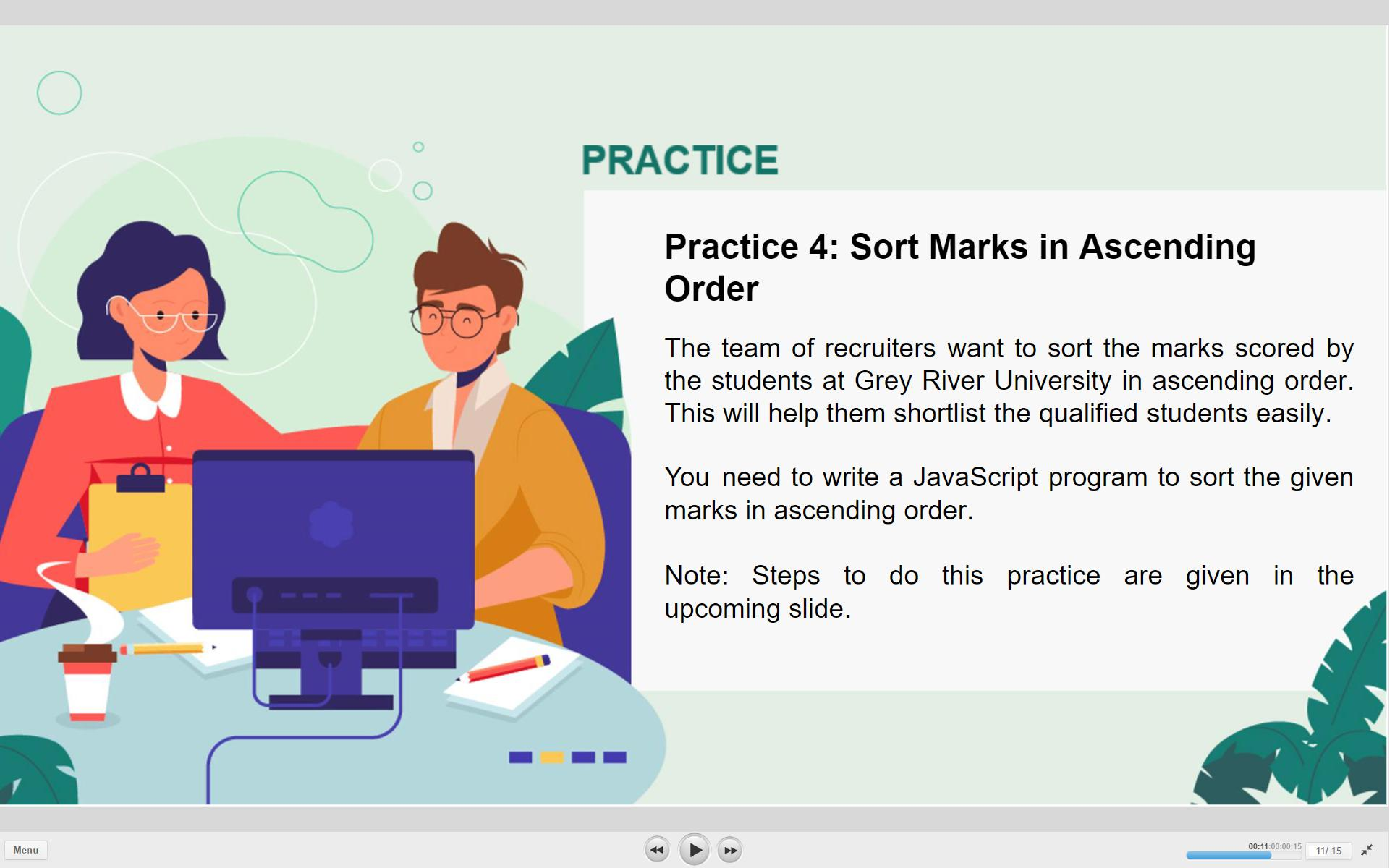
Write a JavaScript program to calculate the count.

Note: Steps to do this practice are given in the upcoming slide.



Practice 3: Tasks

- The solution for this practice should be written inside **p3-submission.js** located inside **p3-count-students** folder.
- Steps to perform this task:
  1. Declare and initialize an array of size 10 with some random marks ranging between 0 and 100.
  2. Initialize a count variable with value 0.
  3. Using a for loop, traverse the array and check whether each value traversed is greater or equal to 40.
  4. Increment the count if the condition evaluates to be true.
  5. Return the count after the loop terminates.



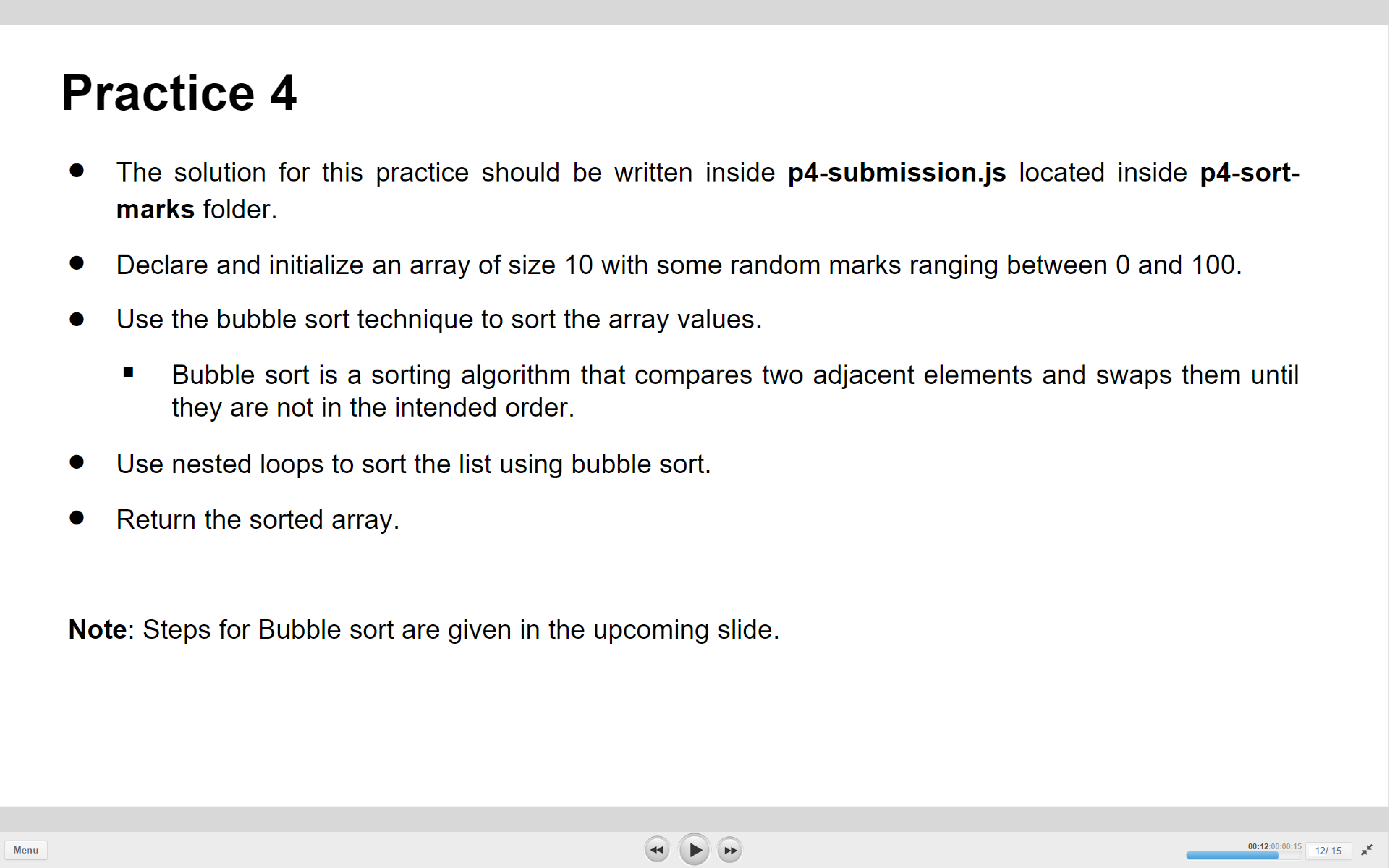
PRACTICE

Practice 4: Sort Marks in Ascending Order

The team of recruiters want to sort the marks scored by the students at Grey River University in ascending order. This will help them shortlist the qualified students easily.

You need to write a JavaScript program to sort the given marks in ascending order.

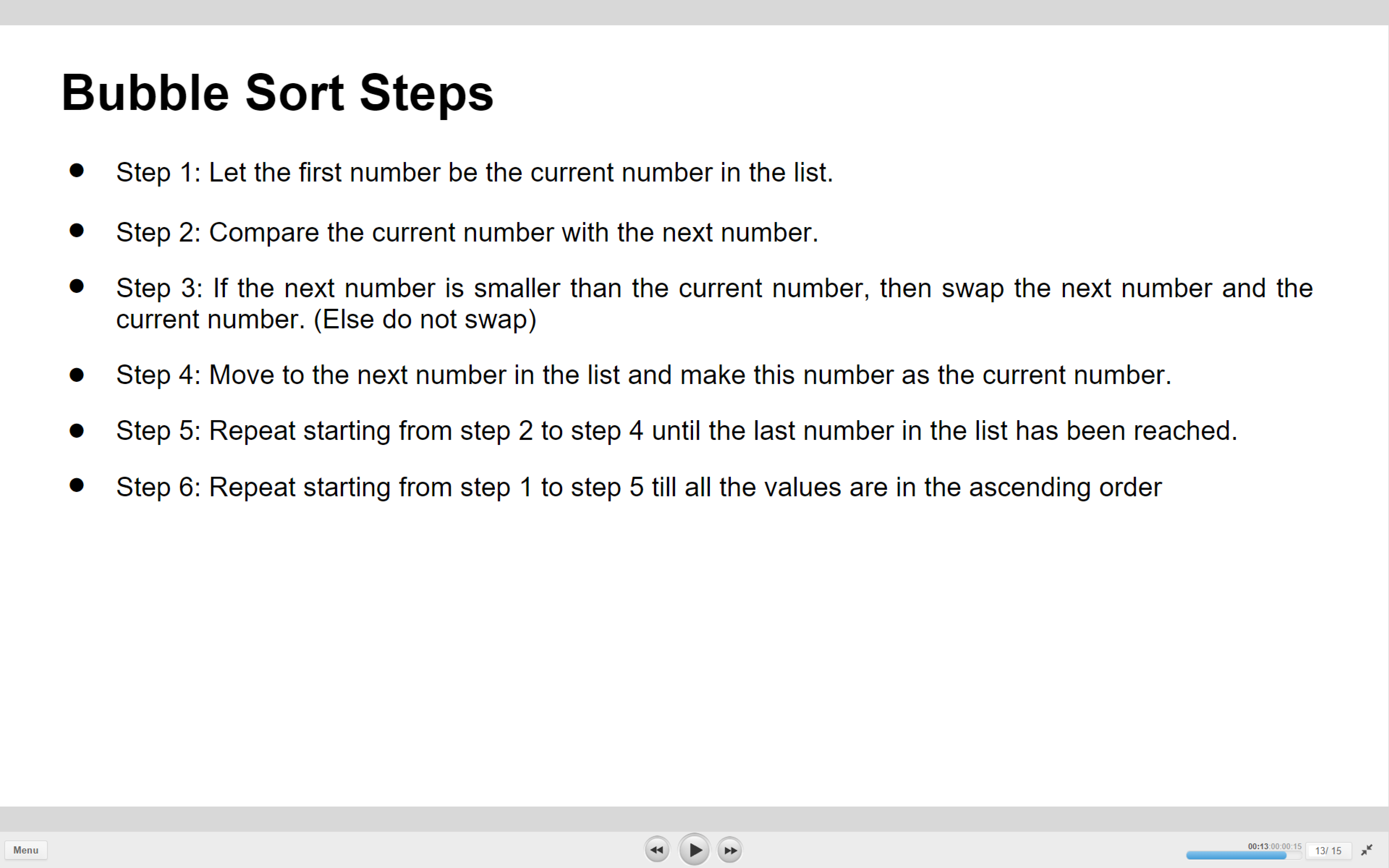
Note: Steps to do this practice are given in the upcoming slide.



Practice 4

- The solution for this practice should be written inside **p4-submission.js** located inside **p4-sort-marks** folder.
- Declare and initialize an array of size 10 with some random marks ranging between 0 and 100.
- Use the bubble sort technique to sort the array values.
  - Bubble sort is a sorting algorithm that compares two adjacent elements and swaps them until they are not in the intended order.
- Use nested loops to sort the list using bubble sort.
- Return the sorted array.

Note: Steps for Bubble sort are given in the upcoming slide.



Bubble Sort Steps

- Step 1: Let the first number be the current number in the list.
- Step 2: Compare the current number with the next number.
- Step 3: If the next number is smaller than the current number, then swap the next number and the current number. (Else do not swap)
- Step 4: Move to the next number in the list and make this number as the current number.
- Step 5: Repeat starting from step 2 to step 4 until the last number in the list has been reached.
- Step 6: Repeat starting from step 1 to step 5 till all the values are in the ascending order