

Assessment Questions

1. Write a function that prints the numbers from 1 to 100. But for multiples of three, print "Fizz" instead of the number, and for the multiples of five, print "Buzz". For numbers that are multiples of both three and five, print "FizzBuzz".

CODE:

```
function fizzBuzz() {  
  for (let i = 1; i <= 100; i++) {  
    if (i % 3 === 0 && i % 5 === 0) {  
      console.log("FizzBuzz");  
    } else if (i % 3 === 0) {  
      console.log("Fizz");  
    } else if (i % 5 === 0) {  
      console.log("Buzz");  
    } else {  
      console.log(i);  
    }  
  }  
}  
  
fizzBuzz();
```

```
index.html ×  script.js ×

1  function fizzBuzz() {
2      for (let i = 1; i <= 100; i++) {
3          if (i % 3 === 0 && i % 5 === 0) {
4              console.log("FizzBuzz");
5          } else if (i % 3 === 0) {
6              console.log("Fizz");
7          } else if (i % 5 === 0) {
8              console.log("Buzz");
9          } else {
10             console.log(i);
11         }
12     }
13 }

14
15 fizzBuzz();
16
17

Console ×
26
Fizz
28
29
FizzBuzz
31
```

2. Write a function that takes a string input representing a simple arithmetic expression (only addition and subtraction) and returns the result.

CODE:

```
function evaluateExpression(expression) {
```

```
    return eval(expression);  
  }  
}
```

```
console.log(evaluateExpression("3 + 5 - 2"));
```



The screenshot shows a code editor with two tabs: 'index.html' and 'script.js'. The 'script.js' tab is active, displaying the following code:

```
1  function evaluateExpression(expression) { // D  
2    return eval(expression);           // U  
3  }  
4  
5  console.log(evaluateExpression("3 + 5 - 2")); /  
6
```

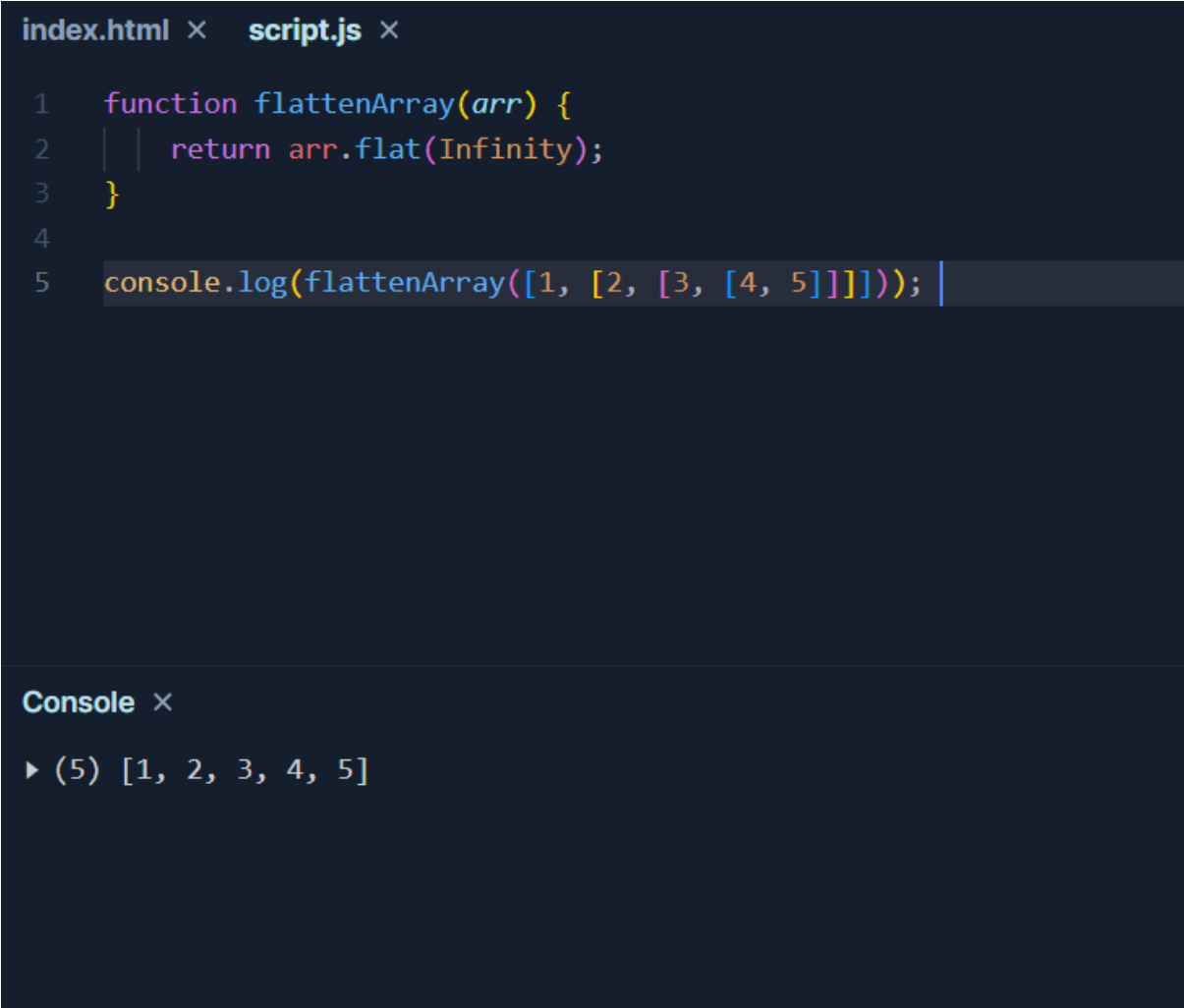
Below the code editor is a 'Console' panel. It shows the output of the code execution, which is the number '6'.

3. Write a function that takes a nested array and returns a flattened array.

CODE:

```
function flattenArray(arr) {  
  
    return arr.flat(Infinity);  
  
}
```

```
console.log(flattenArray([1, [2, [3, [4, 5]]]]));
```



The screenshot shows a code editor with two tabs: 'index.html' and 'script.js'. The 'script.js' tab is active, displaying a function `flattenArray(arr)` that returns `arr.flat(Infinity)`. Below the function, a `console.log` statement calls `flattenArray` with a nested array `[1, [2, [3, [4, 5]]]]`. The console output shows the flattened array `[1, 2, 3, 4, 5]`.

```
index.html ×  script.js ×  
  
1  function flattenArray(arr) {  
2    |    return arr.flat(Infinity);  
3  }  
4  
5  console.log(flattenArray([1, [2, [3, [4, 5]]]])); |  
  
Console ×  
▶ (5) [1, 2, 3, 4, 5]
```

4. Write a function that checks if two given strings are anagrams of each other.

CODE:

```
function areAnagrams(str1, str2) {  
  
    const normalize = str => str.split("").sort().join("");  
  
    return normalize(str1) === normalize(str2);  
  
}
```

```
console.log(areAnagrams("listen", "silent"));
```



The screenshot shows a code editor with two tabs: 'index.html' and 'script.js'. The 'script.js' tab is active, displaying a JavaScript function 'areAnagrams' and a console log call. The function 'areAnagrams' takes two strings, 'str1' and 'str2', and returns a boolean value based on whether the sorted characters of both strings are identical. The console log call 'console.log(areAnagrams("listen", "silent"));' is highlighted. Below the code editor, the 'Console' tab is open, showing the output 'true'.

```
index.html ×  script.js ×  
  
1  function areAnagrams(str1, str2) {  
2      |      const normalize = str => str.split('').sort().join('');  
3      |      return normalize(str1) === normalize(str2);  
4  }  
5  
6  console.log(areAnagrams("listen", "silent")); |  
  
Console ×  
  
true
```

5. Write a function that takes an array and returns a new array with duplicates removed.

CODE:

```
function removeDuplicates(arr) {  
  
    return [...new Set(arr)];  
  
}
```

```
console.log(removeDuplicates([1, 2, 2, 3, 4, 4, 5]));
```

```
index.html ×  script.js ×

1  function removeDuplicates(arr) {
2    |    return [...new Set(arr)];
3  }
4
5  console.log(removeDuplicates([1, 2, 2, 3, 4, 4, 5]));

Console ×

▶ (5) [1, 2, 3, 4, 5]
```

6. Write a function that takes a string and capitalizes the first letter of each word in the string.

CODE:

```
function capitalizeWords(str) {

    return str.split(' ').map(word => word.charAt(0).toUpperCase() + word.slice(1)).join(' ');
}

console.log(capitalizeWords("hello world"));
```



The screenshot shows a web browser's developer console with two tabs: 'index.html' and 'script.js'. The 'script.js' tab is active, displaying a JavaScript function `capitalizeWords(str)`. The function splits the input string into words, capitalizes the first letter of each word, and joins them back together. The console output shows 'Hello World'.

```
function capitalizeWords(str) {  
  return str.split(' ').  
    .map(word => word.charAt(0).toUpperCase() + word.slice(1))  
    .join(' ');  
}  
  
console.log(capitalizeWords("hello world"));
```

Console × ... Web View ×
Hello World

7. Write a function that generates the first **n** numbers of the Fibonacci sequence.

CODE:

```
function fibonacci(n) {  
  
  const sequence = [0, 1];  
  
  for (let i = 2; i < n; i++) {  
  
    sequence.push(sequence[i - 1] + sequence[i - 2]);  
  
  }  
  
  return sequence.slice(0, n);  
  
}  
  
console.log(fibonacci(10));
```

```
index.html ×  script.js ×

1  function fibonacci(n) {
2      const sequence = [0, 1];
3      for (let i = 2; i < n; i++) {
4          sequence.push(sequence[i - 1] + sequence[i - 2]);
5      }
6      return sequence.slice(0, n);
7  }
8
9  console.log(fibonacci(10));

Console ×

▶ (10) [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

8. Implement a simple HashMap class with **put**, **get**, and **remove** methods.

CODE:

```
class HashMap {

    constructor() {

        this.map = {};

    }

    put(key, value) {

        this.map[key] = value;

    }

}
```



```
get(key) {  
    return this.map[key];  
}
```

```
remove(key) {  
    delete this.map[key];  
}  
}
```

```
const map = new HashMap();  
map.put("name", "John");  
console.log(map.get("name"));  
map.remove("name");  
console.log(map.get("name"));
```

index.html × script.js ×

```
1  class HashMap {
2      constructor() {
3          this.map = {};
4      }
5
6      put(key, value) {
7          this.map[key] = value;
8      }
9
10     get(key) {
11         return this.map[key];
12     }
13
14     remove(key) {
15         delete this.map[key];
16     }
17 }
18
19 const map = new HashMap();
20 map.put("name", "John");
21 console.log(map.get("name"));
22 map.remove("name");
23 console.log(map.get("name"));
24
```

Console ×

John

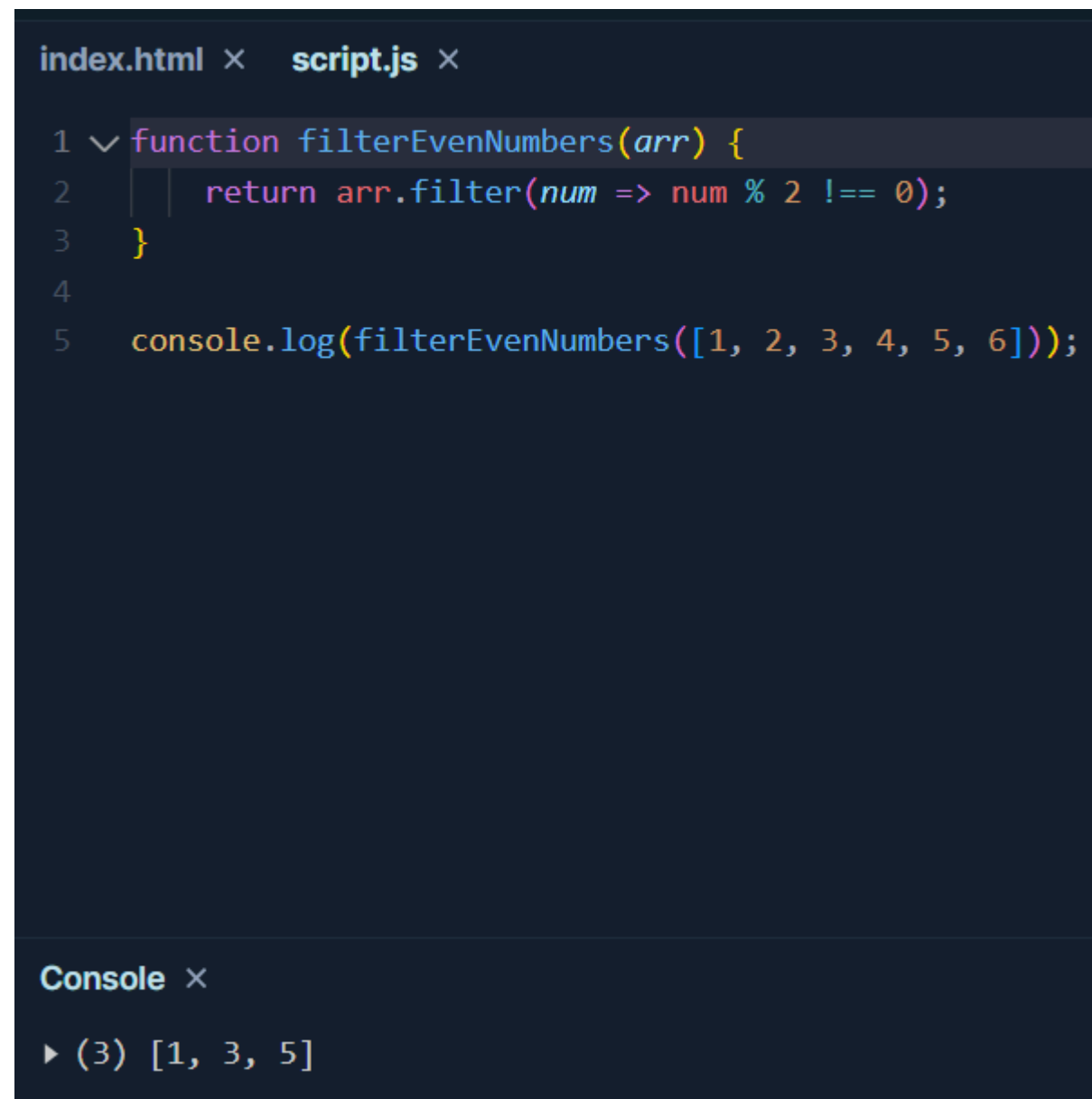
undefined

9. Write a function that filters out even numbers from an array.

CODE:

```
function filterEvenNumbers(arr) {  
  
    return arr.filter(num => num % 2 !== 0);  
  
}
```

```
console.log(filterEvenNumbers([1, 2, 3, 4, 5, 6]));
```



The screenshot shows a code editor with two tabs: 'index.html' and 'script.js'. The 'script.js' tab is active, displaying the following code:

```
1  function filterEvenNumbers(arr) {  
2      return arr.filter(num => num % 2 !== 0);  
3  }  
4  
5  console.log(filterEvenNumbers([1, 2, 3, 4, 5, 6]));
```

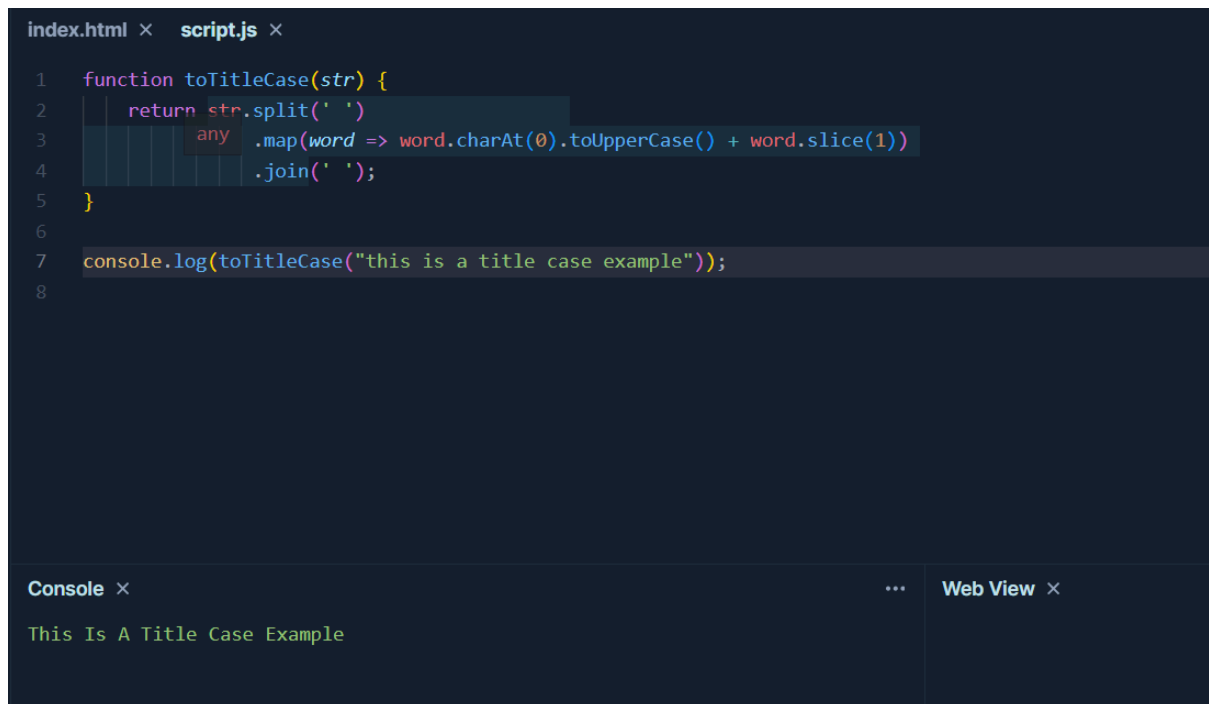
Below the code editor is a 'Console' panel. It shows the output of the code execution: '▶ (3) [1, 3, 5]'. This indicates that the function successfully filtered the even numbers from the array [1, 2, 3, 4, 5, 6], leaving only the odd numbers [1, 3, 5].

10. Write a function that converts a given string to title case (capitalizing the first letter of each word).

CODE:

```
function toTitleCase(str) {  
  
    return str.split(' ').map(word => word.charAt(0).toUpperCase() + word.slice(1)).join(' ');  
  
}
```

```
console.log(toTitleCase("this is a title case example"));
```



The screenshot shows a code editor with two tabs: 'index.html' and 'script.js'. The 'script.js' tab is active, displaying the following JavaScript code:

```
1 function toTitleCase(str) {  
2     return str.split(' ')  
3         .map(word => word.charAt(0).toUpperCase() + word.slice(1))  
4         .join(' ');  
5 }  
6  
7 console.log(toTitleCase("this is a title case example"));  
8
```

Below the code editor, there is a 'Console' tab and a 'Web View' tab. The 'Console' tab is active, showing the output of the code: 'This Is A Title Case Example'.

