

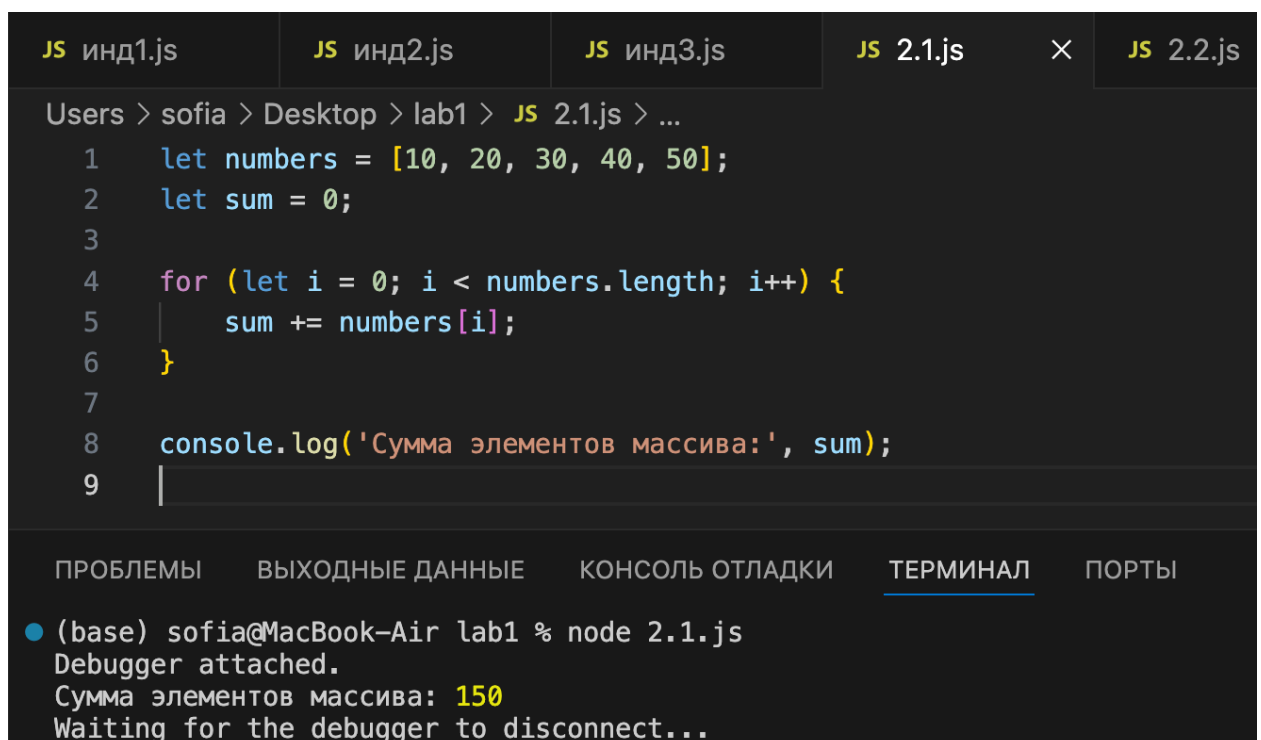
Лабораторная работа 2. Массивы в JavaScript

Цель работы: научиться работать с массивами в JavaScript, использовать основные методы массивов для обработки данных и решения задач.

Задача 1. Найти сумму элементов массива.

Алгоритм:

1. Создала массив с числами.
2. Использовала цикл `for` для вычисления суммы всех элементов массива.
3. Вывела результат на экран.



```
JS инд1.js JS инд2.js JS инд3.js JS 2.1.js × JS 2.2.js
Users > sofia > Desktop > lab1 > JS 2.1.js > ...
1  let numbers = [10, 20, 30, 40, 50];
2  let sum = 0;
3
4  for (let i = 0; i < numbers.length; i++) {
5    |   sum += numbers[i];
6  }
7
8  console.log('Сумма элементов массива:', sum);
9  |

ПРОБЛЕМЫ  ВЫХОДНЫЕ ДАННЫЕ  КОНСОЛЬ ОТЛАДКИ  ТЕРМИНАЛ  ПОРТЫ
● (base) sofia@MacBook-Air lab1 % node 2.1.js
Debugger attached.
Сумма элементов массива: 150
Waiting for the debugger to disconnect...
```

Задача 2. Найти максимальное значение в массиве.

Алгоритм:

1. Создала массив с числами.
2. Использовала цикл `for` для поиска наибольшего элемента массива.
3. Вывела результат на экран.

```
JS инд1.js JS инд2.js JS инд3.js JS 2.1.js JS 2.2.js ×
Users > sofia > Desktop > lab1 > JS 2.2.js > ...
4   for (let i = 1; i < numbers.length; i++) {
5       if (numbers[i] > max) {
6           max = numbers[i];
7       }
8   }
9
10  console.log('Максимальный элемент массива:', max);
11
ПРОБЛЕМЫ Выходные данные Консоль отладки ТЕРМИНАЛ ПОРТЫ
● (base) sofia@MacBook-Air lab1 % node 2.2.js
Debugger attached.
Максимальный элемент массива: 35
Waiting for the debugger to disconnect...
```

Задача 3. Нахождение четных чисел в массиве.

Алгоритм: 1. Создала массив с числами.

2. Использовала метод `filter()`, чтобы получить новый массив, состоящий только из четных чисел.

3. Вывела результат на экран.

```
JS инд1.js JS инд2.js JS инд3.js JS 2.1.js JS 2.2.js JS 2.3.js ×
Users > sofia > Desktop > lab1 > JS 2.3.js > ...
1   let numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
2   let evenNumbers = numbers.filter(num => num % 2 === 0);
3
4   console.log('Чётные числа:', evenNumbers);
5
ПРОБЛЕМЫ Выходные данные Консоль отладки ТЕРМИНАЛ ПОРТЫ
● (base) sofia@MacBook-Air lab1 % node 2.3.js
Debugger attached.
Чётные числа: [ 2, 4, 6, 8, 10 ]
Waiting for the debugger to disconnect...
```

Задача 4. Удаление элементов из массива.

Алгоритм:

1. Создала массив с числами.

2. Использовала метод `splice()` для удаления двух элементов, начиная с индекса 2.

3. Вывела измененный массив на экран.

```
Users > sofia > Desktop > lab1 > JS 2.4.js > ...
1 let numbers = [10, 20, 30, 40, 50, 60];
2 numbers.splice(2, 2); // Удаляет 2 элемента, начиная с индекса 2
3
4 console.log('Изменённый массив:', numbers);
5
```

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ ПОРТЫ

```
• (base) sofia@MacBook-Air lab1 % node 2.4.js
Debugger attached.
Изменённый массив: [ 10, 20, 50, 60 ]
Waiting for the debugger to disconnect...
• (base) sofia@MacBook-Air lab1 % node 2.5.js
```

Задача 5. Преобразование массива с помощью map()

Алгоритм:

1. Создала массив с числами.
2. Использовала метод map(), чтобы умножить каждый элемент массива на 2.
3. Вывела результат на экран.

```
Users > sofia > Desktop > lab1 > JS 2.5.js > ...
1 let numbers = [1, 2, 3, 4, 5];
2 let doubledNumbers = numbers.map(num => num * 2);
3
4 console.log('Массив после умножения на 2:', doubledNumbers);
5
```

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ ПОРТЫ

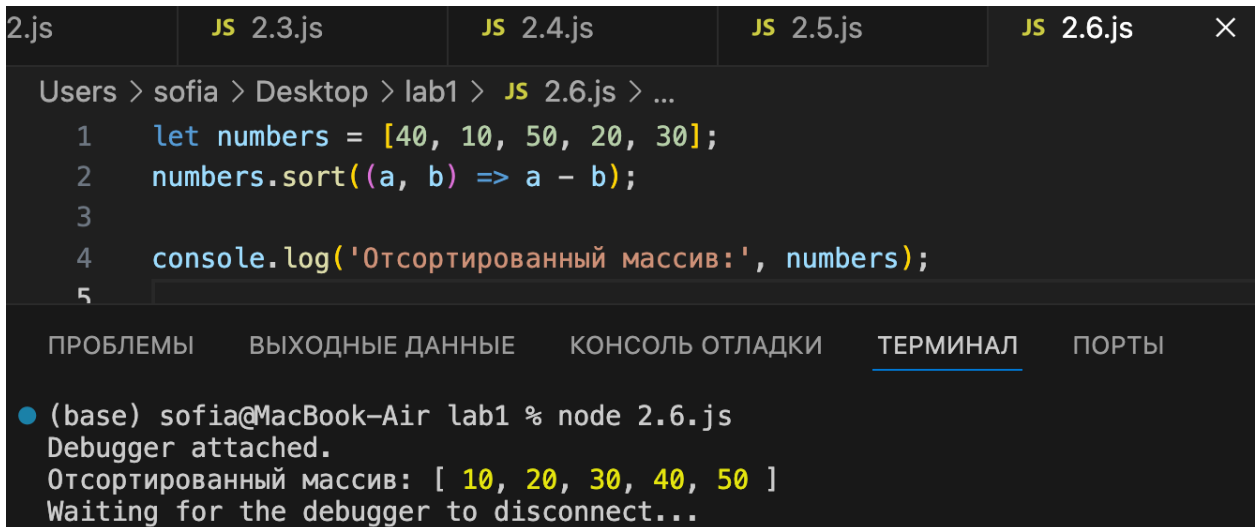
```
• (base) sofia@MacBook-Air lab1 % node 2.5.js
Debugger attached.
Массив после умножения на 2: [ 2, 4, 6, 8, 10 ]
Waiting for the debugger to disconnect...
```

Задача 6. Сортировка массива

Алгоритм:

1. Создала массив с числами.
2. Использовала метод sort(), чтобы отсортировать массив по возрастанию.

3. Вывела результат на экран.



The screenshot shows a code editor with a tab for `2.6.js`. The code in the editor is as follows:

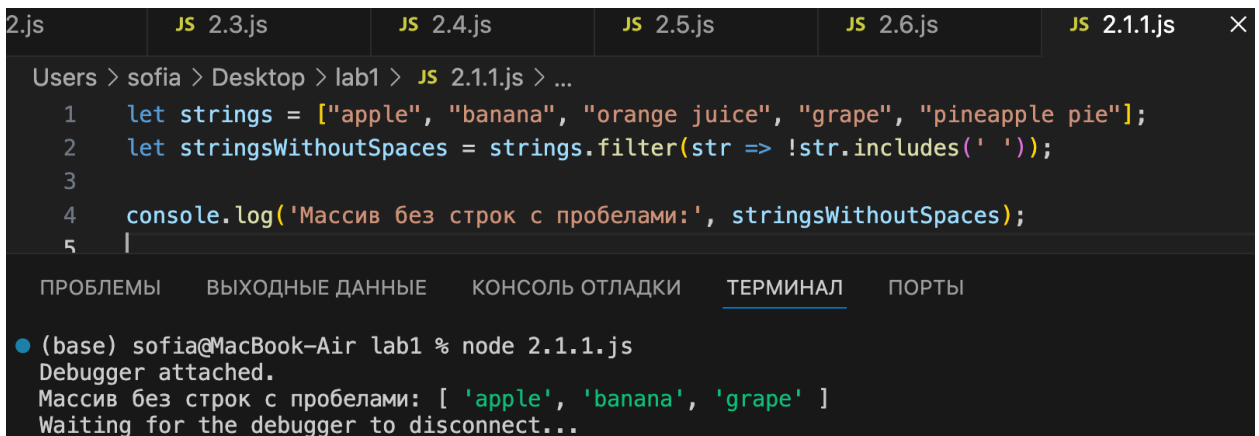
```
1 let numbers = [40, 10, 50, 20, 30];
2 numbers.sort((a, b) => a - b);
3
4 console.log('Отсортированный массив:', numbers);
5
```

Below the code editor, the terminal output is displayed:

```
(base) sofia@MacBook-Air lab1 % node 2.6.js
Debugger attached.
Отсортированный массив: [ 10, 20, 30, 40, 50 ]
Waiting for the debugger to disconnect...
```

Вариант 22

1. Создание массива строк, не содержащих пробелы. Удалите из массива все строки, содержащие пробелы.



The screenshot shows a code editor with a tab for `2.1.1.js`. The code in the editor is as follows:

```
1 let strings = ["apple", "banana", "orange juice", "grape", "pineapple pie"];
2 let stringsWithoutSpaces = strings.filter(str => !str.includes(' '));
3
4 console.log('Массив без строк с пробелами:', stringsWithoutSpaces);
5
```

Below the code editor, the terminal output is displayed:

```
(base) sofia@MacBook-Air lab1 % node 2.1.1.js
Debugger attached.
Массив без строк с пробелами: [ 'apple', 'banana', 'grape' ]
Waiting for the debugger to disconnect...
```

Код создает массив строк и фильтрует его, удаляя все строки, содержащие пробелы, с помощью метода `filter()`. Результирующий массив, в котором остались только строки без пробелов, выводится на экран.

2. Удаление всех чисел, больших N. Удалите из массива все числа, превышающие значение N, введенное пользователем.

```
Users > sofia > Desktop > lab1 > JS 2.1.2.js > ...
1  const readline = require('readline');
2
3  const rl = readline.createInterface({
4      input: process.stdin,
5      output: process.stdout
6  });
7
8  let numbers = [5, 12, 30, 8, 22, 15, 3];
9
10 rl.question('Введите значение N: ', (input) => {
11     let N = parseInt(input); // Преобразуем введенное значение в число
12     let filteredNumbers = numbers.filter(num => num <= N);
13
14     console.log('Массив после удаления чисел, больших N:', filteredNumbers);
15
16     rl.close(); // Закрываем интерфейс
17 });
```

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ ПОРТЫ

```
● (base) sofia@MacBook-Air lab1 % node 2.1.2.js
Debugger attached.
Введите значение N: 15
Массив после удаления чисел, больших N: [ 5, 12, 8, 15, 3 ]
Waiting for the debugger to disconnect...
```

Этот код использует модуль `readline` для взаимодействия с пользователем в среде Node.js. Он запрашивает у пользователя значение `N`, а затем фильтрует массив чисел, удаляя все элементы, превышающие `N`. Результирующий массив выводится на экран, после чего интерфейс закрывается.

3. Подсчет количества элементов массива, которые начинаются с определенной буквы. Найдите количество строк в массиве, которые начинаются с указанной буквы.

```
Users > sofia > Desktop > lab1 > JS 2.1.3.js > ...
1  const readline = require('readline');
2
3  const rl = readline.createInterface({
4      input: process.stdin,
5      output: process.stdout
6  });
7
8  let words = ["apple", "banana", "apricot", "grape", "avocado", "berry"];
9
10 rl.question('Введите букву для поиска: ', (input) => {
11     let letter = input.toLowerCase(); // Преобразуем введенную букву в нижний регистр
12
13     // Фильтруем строки, которые начинаются с введенной буквы
14     let count = words.filter(word => word.toLowerCase().startsWith(letter)).length;
15
16     console.log(`Количество строк, начинающихся с буквы "${letter}":`, count);
17
18     rl.close(); // Закрываем интерфейс
19 });
20
```

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ

✓ **ТЕРМИНАЛ**

```
● (base) sofia@MacBook-Air lab1 % node 2.1.3.js
Debugger attached.
Введите букву для поиска: b
Количество строк, начинающихся с буквы "b": 2
Waiting for the debugger to disconnect...
```

Этот код использует модуль `readline` для взаимодействия с пользователем в среде Node.js. Он запрашивает у пользователя букву, с которой должны начинаться строки, и затем фильтрует массив слов, подсчитывая количество строк, начинающихся с этой буквы. Результат выводится на экран, а затем интерфейс закрывается.