**федеральное государственное автономное образовательное**



**учреждение высшего образования**

**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**(ВЫСШАЯ ШКОЛА ПЕЧАТИ И МЕДИАИНДУСТРИИ)**

**(Факультет информационных технологий)**

***(Институт Принтмедиа и информационных технологий) Кафедра Информатики и информационных технологий***

**направление подготовки 09.03.02 «Информационные системы и технологии»**

# ЛАБОРАТОРНАЯ РАБОТА № 1

**Дисциплина: Функциональное программирование.**

**Тема: Основы JS для функционального программирования.**

**Выполнил(а): студент(ка) группы 221-3711**

**Мироненко Р. Е.**

## (Фамилия И.О.)

**Дата, подпись** **13.02.2025**  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

## **Проверил: \_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_***

(Фамилия И.О., степень, звание) **(Оценка)**

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва2025**

**Цель:** Освоить основные принципы функционального программирования в JavaScript, научиться создавать чистые функции и использовать функции высшего порядка.

**Задание:**

1. **Разработайте набор чистых функций для работы с массивами:**

* Функция, которая принимает массив чисел и возвращает новый массив, содержащий только четные числа.
* Функция, которая принимает массив чисел и возвращает новый массив, содержащий квадраты этих чисел.
* Функция, которая принимает массив объектов и возвращает новый массив, содержащий только объекты с определенным свойством.
* Функция, которая принимает массив чисел и возвращает их сумму.

1. **Создайте функцию высшего порядка, которая принимает функцию и массив в качестве аргументов и применяет функцию к каждому элементу массива, возвращая новый массив с результатами.**
2. **Используя разработанные функции, выполните следующие математические операции:**

* Найдите сумму квадратов всех чётных чисел в заданном массиве.
* Найдите среднее арифметическое всех чисел, больших заданного значения, в заданном массиве объектов.

**Требования:**

* Все функции должны быть чистыми, то есть не иметь побочных эффектов и всегда возвращать одинаковый результат при одинаковых аргументах.
* Используйте стрелочные функции и другие современные возможности JavaScript.
* Код должен быть хорошо оформлен и легко читаем.

**Ход работы:**

Гит: <https://github.com/Roman784/FuncProg.git>

Листинг 1 lab1\_main.js

|  |
| --- |
| // Функции ------------------------------------------------------------------------------------------------------  // Функция, которая принимает массив чисел и возвращает новый массив, содержащий только четные числа.  *const* getEvenNumbers = (arr) *=>* arr.filter(num *=>* num % 2 === 0);  // Функция, которая принимает массив чисел и возвращает новый массив, содержащий квадраты этих чисел.  *const* getSquares = (arr) *=>* arr.map(num *=>* num \*\* 2);  // Функция, которая принимает массив объектов и возвращает новый массив, содержащий только объекты с определенным свойством.  *const* filterByProperty = (arr, prop) *=>* arr.filter(obj *=>* obj.hasOwnProperty(prop));  // Функция, которая принимает массив чисел и возвращает их сумму.  *const* getSum = (arr) *=>* arr.reduce((sum, num) *=>* sum + num, 0);  // Функция высшего порядка, которая принимает функцию и массив в качестве аргументов и применяет функцию к каждому элементу массива, возвращая новый массив с результатами.  *const* applyFunction = (func, arr) *=>* arr.map(func);  // Поиск среднего арифметического.  *const* getAverage = (arr) *=>* getSum(arr) / arr.length;  // Демонстрация работы ------------------------------------------------------------------------------------------  *const* numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9];  *const* objects = [{num: 1}, {obj: *null*}, {num: 2}, {value: "value1"}, {value: "value2"}, {num: 3}, {num: 4}];  console.log("Исходный массив чисел: " + numbers);  console.log("Исодный массив объектов: " + JSON.stringify(objects) + "\n");  console.log("Чётные числа: " + getEvenNumbers(numbers));  console.log("Квадраты чисел: " + getSquares(numbers));  console.log("Объекты с определённым свойством (num): " + JSON.stringify(filterByProperty(objects, "num")));  console.log("Сумма чисел: " + getSum(numbers));  console.log("Удвоение каждого элемента: " + applyFunction(num *=>* num \* 2, numbers) + "\n");  console.log("Сумма квадратов чётных чисел: " + getSum(getSquares(getEvenNumbers(numbers))));  console.log("Среднее арифметическое чисел в массиве объектов больше 2: " +      getAverage(          applyFunction(obj *=>* obj["num"], filterByProperty(objects, "num")) // Собираем числа из объектов в отдельный массив.          .filter(value *=>* value > 2) // Отбираем числа больше 2.      )); |

Для решения последней задачи с поиском среднего арифметического, я написал вспомогательную функцию getAverage.

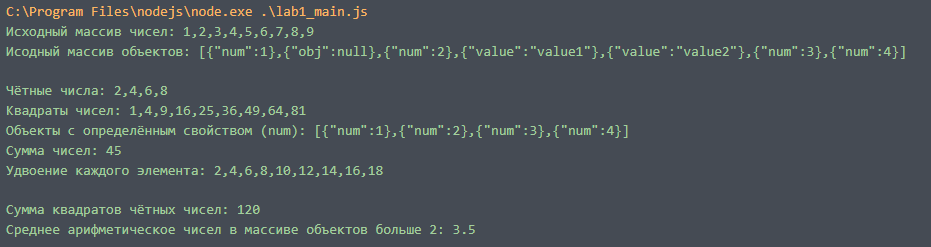


Рисунок 1 Результат работы в консоли