ГУАП

КАФЕДРА № 43

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ

ПРАКТИЧЕСКОЕ ЗАДАНИЕ №3

по дисциплине: JAVASCRIPT, ЕГО БИБЛИОТЕКИ И ФРЕЙМВОРКИ В FRONTEND-РАЗРАБОТКЕ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР.	4232		Г. П. Матюшков
		подпись, дата	инициалы, фамилия

Санкт-Петербург
2025

Цель работы:

Формирование практических навыков работы с асинхронным кодом в языке JavaScript, используя callback-функции и объект Promise.

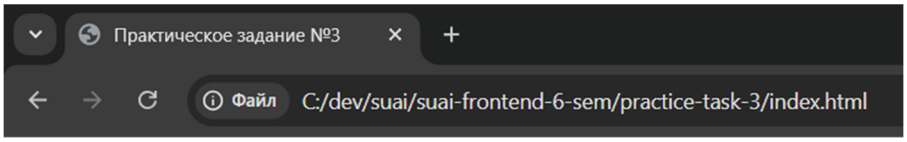
Задание:

1. Напишите функцию `filterArray`, которая принимает массив чисел и callback-функцию. Функция `filterArray` должна вызывать callback функцию для каждого элемента массива и возвращать новый массив, содержащий только те элементы, для которых callback-функция вернула `true`.

Напишите 2 примера применения этой функции. К примеру, для фильтрации четных и нечетных значений массива.

2. Напишите асинхронную функцию `fetchData`, которая принимает URL в качестве параметра и возвращает Promise. Функция должна использовать `fetch` для получения данных с указанного URL. Если запрос прошел успешно (статус ответа 200), Promise должен быть разрешен с полученными данными в виде строки. Если запрос не удался (любой другой статус), Promise должен быть отклонен с сообщением об ошибке.

Результаты работы:



Практическое задание №3

Результаты работы программы можно увидеть в консоли разработчика **F12**

Рисунок 1 – главная страница

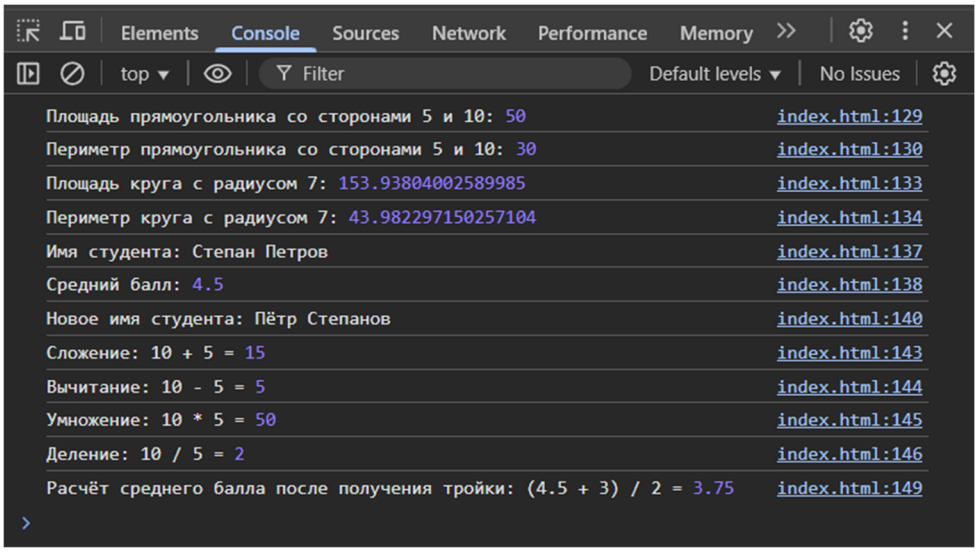


Рисунок 2 – результаты работы в консоли разработчика

Листинг кода:

index.html:

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Практическое задание №3</title>
</head>
<body>
    <h1>Практическое задание №3</h1>
    <p>Результаты работы программы можно увидеть в консоли разработчика
<strong>F12</strong></p>
    <script>
        // Класс Фигура
        class Shape {
            // Метод для вычисления площади (будет переопределен в дочерних
классах)
            getArea() {
                throw new Error("Метод getArea должен быть переопределен");
            }

            // Метод для вычисления периметра (будет переопределен в дочерних
классах)
            getPerimeter() {
                throw new Error("Метод getPerimeter должен быть переопределен");
            }
        }

        // Класс Прямоугольник, наследуется от Фигуры
        class Rectangle extends Shape {
            constructor(width, height) {
                super();
                this.width = width;
                this.height = height;
            }

            // Переопределение метода для вычисления площади
            getArea() {
                return this.width * this.height;
            }

            // Переопределение метода для вычисления периметра
            getPerimeter() {
                return 2 * (this.width + this.height);
            }
        }

        // Класс Круг, наследуется от Фигуры
        class Circle extends Shape {
            constructor(radius) {
                super();
                this.radius = radius;
            }

            // Переопределение метода для вычисления площади
```

```
    getArea() {
        return Math.PI * this.radius ** 2;
    }

    // Переопределение метода для вычисления периметра
    getPerimeter() {
        return 2 * Math.PI * this.radius;
    }
}

// Класс Студент
class Student {
    constructor(name, age, averageGrade) {
        this._name = name;
        this._age = age;
        this._averageGrade = averageGrade;
    }

    // Геттер для имени
    get name() {
        return this._name;
    }

    // Сеттер для имени
    set name(newName) {
        this._name = newName;
    }

    // Геттер для возраста
    get age() {
        return this._age;
    }

    // Сеттер для возраста
    set age(newAge) {
        this._age = newAge;
    }

    // Геттер для среднего балла
    get averageGrade() {
        return this._averageGrade;
    }

    // Сеттер для среднего балла
    set averageGrade(newAverageGrade) {
        this._averageGrade = newAverageGrade;
    }
}

// Класс Калькулятор
class Calculator {
    // Метод для сложения
    add(a, b) {
        return a + b;
    }

    // Метод для вычитания
    subtract(a, b) {
        return a - b;
```

```
            }

            // Метод для умножения
            multiply(a, b) {
                return a * b;
            }

            // Метод для деления
            divide(a, b) {
                if (b === 0) {
                    throw new Error("Деление на ноль невозможно");
                }
                return a / b;
            }
        }

        // Пример использования классов
        const rectangle = new Rectangle(5, 10);
        console.log("Площадь прямоугольника со сторонами 5 и 10:",
rectangle.getArea());
        console.log("Периметр прямоугольника со сторонами 5 и 10:",
rectangle.getPerimeter());

        const circle = new Circle(7);
        console.log("Площадь круга с радиусом 7:", circle.getArea());
        console.log("Периметр круга с радиусом 7:", circle.getPerimeter());

        const student = new Student("Степан Петров", 20, 4.5);
        console.log("Имя студента:", student.name);
        console.log("Средний балл:", student.averageGrade);
        student.name = "Пётр Степанов";
        console.log("Новое имя студента:", student.name);

        const calculator = new Calculator();
        console.log("Сложение: 10 + 5 =", calculator.add(10, 5));
        console.log("Вычитание: 10 - 5 =", calculator.subtract(10, 5));
        console.log("Умножение: 10 * 5 =", calculator.multiply(10, 5));
        console.log("Деление: 10 / 5 =", calculator.divide(10, 5));

        student.averageGrade =
calculator.divide(calculator.add(student.averageGrade, 3), 2);
        console.log("Расчёт среднего балла после получения тройки: (4.5 + 3) / 2
=", student.averageGrade);
    </script>
</body>
</html>
```

Вывод:

В ходе выполнения лабораторной работы были успешно реализованы классы на языке JavaScript, демонстрирующие принципы объектно-ориентированного программирования (ООП). Были созданы классы Shape, Rectangle, Circle, Student и Calculator, которые иллюстрируют наследование, инкапсуляцию и полиморфизм. Классы Rectangle и Circle наследуют методы getArea и getPerimeter от базового класса Shape, переопределяя их для вычисления площади и периметра соответствующих фигур. Класс Student демонстрирует

использование геттеров и сеттеров для управления свойствами объекта. Класс Calculator реализует базовые арифметические операции. Результаты работы классов были выведены в консоль разработчика, что подтверждает корректность их реализации. Работа показала умение применять ООП-принципы в JavaScript, а также навыки работы с классами, методами и свойствами. Все задачи выполнены в соответствии с требованиями, результаты работы наглядно представлены и корректны.