МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ

(КАФЕДРА 43)

ОЦЕНКА

ПРЕПОДАВАТЕЛЬ

Старший преподаватель		Н.А. Соловьева
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3

«Основы JavaScript»

по дисциплине: Web-технологии

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №	4236		Л. Мвале
	номер группы	подпись, дата	инициалы, фамилия

Санкт-Петербург 2025

СОДЕРЖАНИЕ

Цель работы: знакомство с языком javascript

1 Варианты задания

Таблица 1 - Вариант задания

№	Гр 4236	Вариант
21	Лисон	5

Вариант задания (базовая часть)

5) Из матрицы размером n на m удалить строки, содержащие более трех отрицательных элементов.

Вариант задания (Расширенная часть)

Нарисовать заданную фигуру, используя скрипт. Повторяющие фрагменты формировать с помощью циклов.

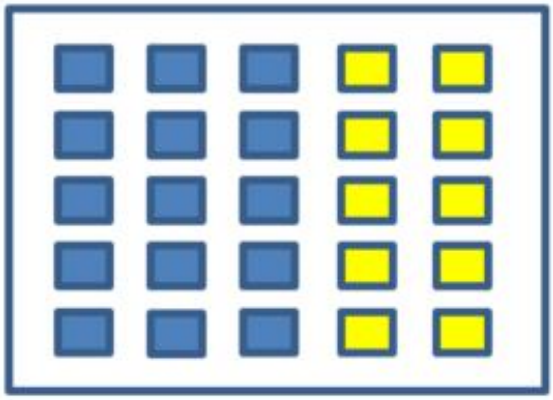


Рисунок 1 – Вариант задания (расширенная часть)

2 Таблицы с описанием переменных программ

Таблица 2 – Описание переменных программы базовой части

Имя переменной	Тип данных	назначение
N	number	Задается число строк в матрице
M	number	Задается число стобцов в матрице
matrix	object	Двумерный массив

choice	number	число, позволяющее определить, следует ли нам заполнять матрицу автоматически или вручную
i	number	Счетчик строк в матрице
j	number	Счетчик столбцов в матрице

Таблица 3 – Описание переменных программы расширенной части

Имя переменной	**Тип данных**	**Назначение**
canvas	HTMLCanvasElement	Ссылка на HTML-элемент <canvas> для рисования.
ctx	CanvasRenderingContext2D	2D-контекст для выполнения графических операций на холсте.
row	number	Счетчик цикла, обозначающий текущую строку сетки (0–4).
col	number	Счетчик цикла, обозначающий текущий столбец сетки (0–4).
x	number	Координата X верхнего левого угла квадрата в сетке.
y	number	Координата Y верхнего левого угла квадрата в сетке.
color	string	Цвет заполнения квадрата ("blue" или "yellow"), в зависимости от колонки.
drawSquare(x, y, color)	function	Функция для рисования квадрата с заданными координатами и цветом.

3 Описание использованных методов массива

Название метода	Параметры	Назначение
map()	callbackFunction(element, index, array)	Создает новый массив, применяя функцию к каждому элементу исходного массива. Используется для форматирования матрицы перед отображением.
filter()	callbackFunction(element, index, array)	Создает новый массив, содержащий только элементы, удовлетворяющие заданному условию. Используется для удаления строк с более чем 3 отрицательными числами.
forEach()	callbackFunction(element, index, array)	Выполняет итерацию по элементам массива без возврата нового массива. Используется для обработки данных перед выводом.
split()	разделитель (необязательный)	Разделяет строку на массив на основе заданного разделителя. Используется для преобразования ввода пользователя в массив чисел.
map(Number)	нет	Преобразует элементы массива из строк в числа. Используется для корректной обработки данных.

4 Описание использованных методов других стандартных объектов

Название метода	Параметры	Назначение
getElementById()	id (строка)	Получает HTML-элемент по его ID. Используется для обращения к <textarea> для ввода и вывода.
prompt()	сообщение (строка)	Показывает диалоговое окно с запросом ввода данных. Используется для запроса размеров матрицы и значений.

alert()	сообщение (строка)	Показывает всплывающее сообщение. Используется для уведомления пользователей об ошибках.
eval()	выражение (строка)	Выполняет JavaScript-код, переданный в виде строки. Используется для исполнения пользовательского скрипта.
padStart()	targetLength (число), padString (строка)	Добавляет пробелы или символы в начало строки для выравнивания. Используется для форматирования матрицы.

5 Текст программ на javascript

Текст программы базовой части:

```
// Шаг 1: Получить размер матрицы от пользователя
let n = parseInt(prompt("Введите количество строк (n):"));
let m = parseInt(prompt("Введите количество столбцов (m):"));

if (isNaN(n) || isNaN(m) || n <= 0 || m <= 0) {
   alert("❌ Пожалуйста, введите корректные положительные числа.");
} else {
   let matrix = [];

   // Шаг 2: Спросить пользователя, хочет ли он вводить значения вручную или заполнить автоматически
   let choice = prompt("Введите '1', чтобы ввести значения вручную, или '2', чтобы заполнить случайными числами:");

   if (choice === "1") {
      // Ввод вручную
      for (let i = 0; i < n; i++) {
```

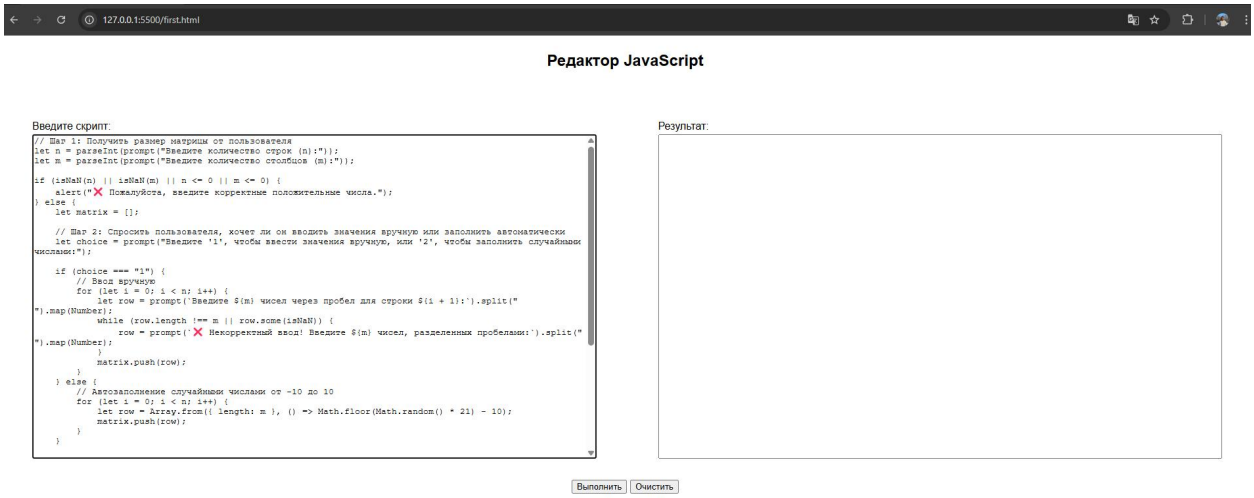
```
        let row = prompt(`Введите ${m} чисел через пробел для строки ${i + 1}:`).split(" ").map(Number);
        while (row.length !== m || row.some(isNaN)) {
            row = prompt(`❌ Некорректный ввод! Введите ${m} чисел, разделенных пробелами:`).split(" ").map(Number);
        }
        matrix.push(row);
    }
} else {
    // Автозаполнение случайными числами от -10 до 10
    for (let i = 0; i < n; i++) {
        let row = Array.from({ length: m }, () => Math.floor(Math.random() * 21) - 10);
        matrix.push(row);
    }
}

// Шаг 3: Форматирование исходной матрицы для отображения
let originalMatrixText = "\n    **Исходная матрица:**\n" +
    matrix.map(row => row.map(num => num.toString().padStart(4)).join(" ")).join("\n");

// Шаг 4: Удаление строк с более чем 3 отрицательными числами
let filteredMatrix = matrix.filter(row => row.filter(num => num < 0).length <= 3);

// Шаг 5: Форматирование отфильтрованной матрицы для отображения
let filteredMatrixText = filteredMatrix.length
    ? "\n    **Отфильтрованная матрица (удалены строки с более чем 3 отрицательными числами):**\n" +
      filteredMatrix.map(row => row.map(num => num.toString().padStart(4)).join(" ")).join("\n")
    : "❌ Не осталось допустимых строк.";

// Шаг 6: Отобразить результат в поле вывода
```

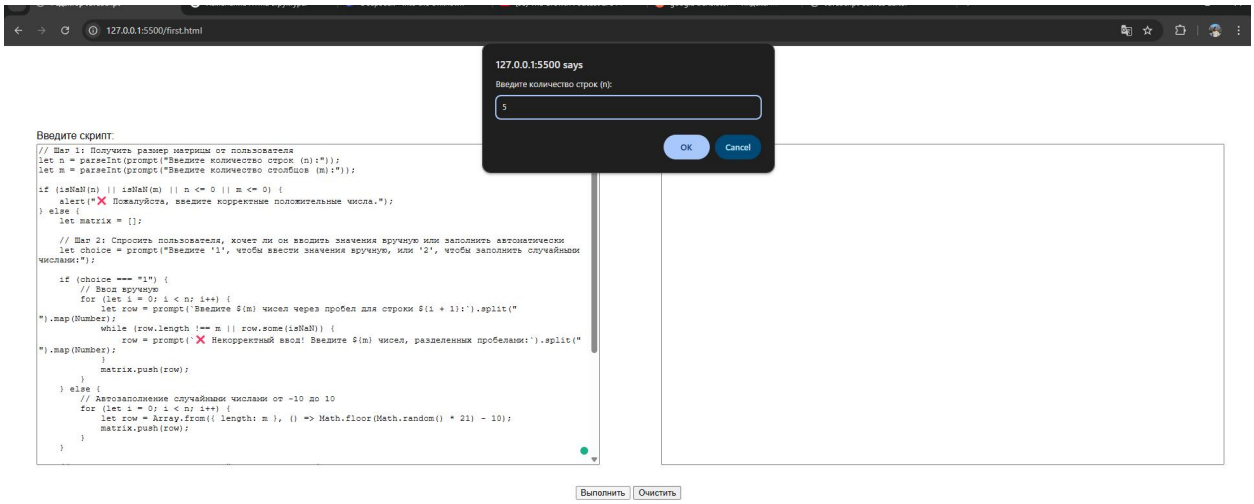
```
    document.getElementById('myrezult').value =
`${originalMatrixText}\n\n${filteredMatrixText}`;
}
```

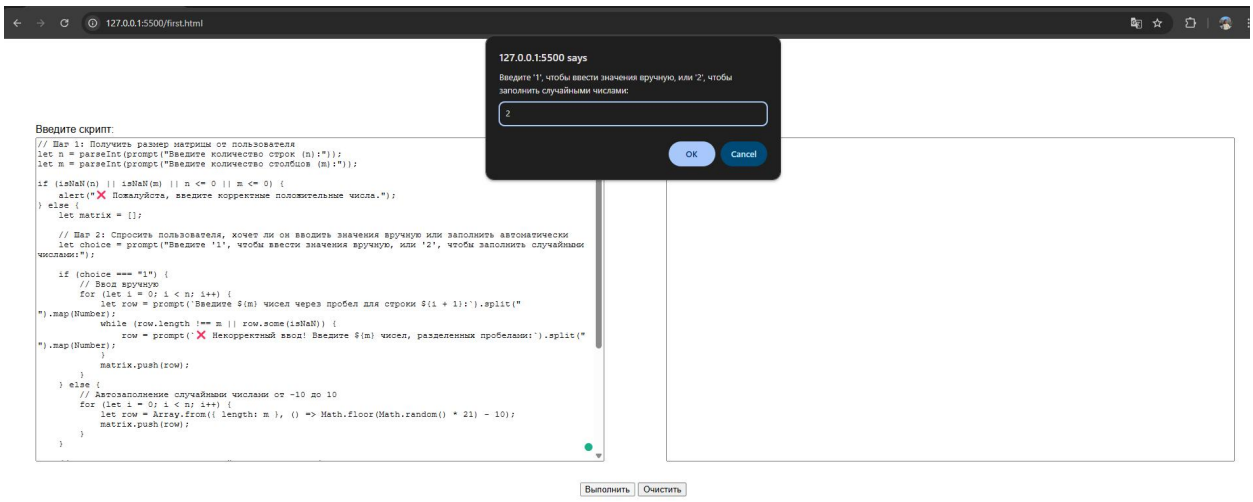
Текст программы расширенной части:

```
let canvas = document.getElementById("drawingCanvas");
let ctx = canvas.getContext("2d");

// Draw the outer blue-edged rectangle
ctx.strokeStyle = "darkblue";
ctx.lineWidth = 4;
ctx.strokeRect(30, 30, 320, 320); // Adjusted size for more space around the grid

// Function to draw an individual square with thicker dark blue borders and spacing
function drawSquare(x, y, color) {
    ctx.fillStyle = color;
    ctx.fillRect(x, y, 40, 40); // Smaller squares (40x40)
    ctx.strokeStyle = "darkblue"; // Dark blue for the border
    ctx.lineWidth = 4; // Thicker border
    ctx.strokeRect(x, y, 40, 40); // Draw square border
}

// Loop to draw the 5x5 grid with larger spaces between squares
for (let row = 0; row < 5; row++) {
    for (let col = 0; col < 5; col++) {
        let x = 50 + col * 60; // Increased spacing between columns
        let y = 50 + row * 60; // Increased spacing between rows
        let color = col < 3 ? "blue" : "yellow"; // Choose color based on column
        drawSquare(x, y, color);
    }
}
```

6 Скриншоты результатов выполнения программ







Результат:

```
📌 **Исходная матрица:**
 -3   10    3    0   -9   -7
  1    6   -5   10    7   -2
 -8    9   -2    6    4    3
 -7   -8    2   -1    6    8
 -3  -10    5   -1    4   -2


📌 **Отфильтрованная матрица (удалены строки с более чем 3 отрицательными числами):**
 -3   10    3    0   -9   -7
  1    6   -5   10    7   -2
 -8    9   -2    6    4    3
 -7   -8    2   -1    6    8
```

Результат:

```
📌 **Исходная матрица:**
 -6   -7   -5   -4
 -7   -6   -6   -7

❌ Не осталось допустимых строк.
```

Введите скрипт:

```
            let row = prompt(`Введите ${m} чисел через пробел для строки ${i + 1}:`).split("
").map(Number);
            while (row.length !== m || row.some(isNaN)) {
                row = prompt(`❌ Некорректный ввод! Введите ${m} чисел, разделенных пробелами:`).split("
").map(Number);
            }
            matrix.push(row);
        }
    } else {
        // Автозаполнение случайными числами от -10 до 10
        for (let i = 0; i < n; i++) {
            let row = Array.from({ length: m }, () => Math.floor(Math.random() * 21) - 10);
            matrix.push(row);
        }
    }

    // Шаг 3: Форматирование исходной матрицы для отображения
    let originalMatrixText = "\n📌 **Исходная матрица:**\n" +
        matrix.map(row => row.map(num => num.toString().padStart(4)).join(" ")).join("\n");

    // Шаг 4: Удаление строк с более чем 3 отрицательными числами
    let filteredMatrix = matrix.filter(row => row.filter(num => num < 0).length <= 3);

    // Шаг 5: Форматирование отфильтрованной матрицы для отображения
    let filteredMatrixText = filteredMatrix.length
        ? "\n📌 **Отфильтрованная матрица (удалены строки с более чем 3 отрицательными числами):**\n" +
          filteredMatrix.map(row => row.map(num => num.toString().padStart(4)).join(" ")).join("\n")
        : "❌ Не осталось допустимых строк.";

    // Шаг 6: Отобразить результат в поле вывода
    document.getElementById('myrezult').value = `${originalMatrixText}\n\n${filteredMatrixText}`;
}
```

Расширенное задание:

Редактор JavaScript Canvas

Введите код JavaScript:

```
let canvas = document.getElementById("drawingCanvas");
let ctx = canvas.getContext("2d");

// Draw the outer blue-edged rectangle
ctx.strokeStyle = "darkblue";
ctx.lineWidth = 4;
ctx.strokeRect(30, 30, 320, 320); // Adjusted size for more space around the
grid

// Function to draw an individual square with thicker dark blue borders and
spacing
function drawSquare(x, y, color) {
    ctx.fillStyle = color;
    ctx.fillRect(x, y, 40, 40); // Smaller squares (40x40)
    ctx.strokeStyle = "darkblue"; // Dark blue for the border
    ctx.lineWidth = 4; // Thicker border
    ctx.strokeRect(x, y, 40, 40); // Draw square border
}

// Loop to draw the 5x5 grid with larger spaces between squares
for (let row = 0; row < 5; row++) {
    for (let col = 0; col < 5; col++) {
        let x = 50 + col * 60; // Increased spacing between columns
        let y = 50 + row * 60; // Increased spacing between rows
        let color = col < 3 ? "blue" : "yellow"; // Choose color based on column
        drawSquare(x, y, color);
    }
}
```

Выполнить код | Чистый холст

Вывод холста:

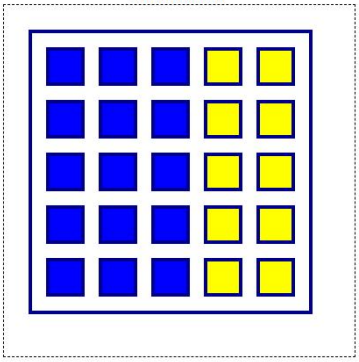


Рисунок 8 – Демонстрация работы расширенного задания

7 Анализ программы и результатов её работы

Назначение программы

Программа представляет собой редактор JavaScript-кода, где пользователи могут вводить код, запускать его и просматривать результат. Выполнение кода происходит внутри элемента <textarea>, а также есть возможность очистки скрипта и результата.

Основные функции

- Обработка ввода пользователя: Вводится JavaScript-код, который выполняется через eval().
- Обработка ошибок: В случае некорректных данных отображаются alert()-уведомления.
- Операции с матрицами: Генерация, фильтрация и форматирование матриц с использованием методов массивов.
- Удобный интерфейс: Отдельные области для ввода кода и вывода результатов повышают удобство работы.

Вывод:

В ходе выполнения лабораторной работы были созданы 2 программы на языке javascript по заданию из базовой и расширенной части. Программа из базовой части была спроектирована с учетом различных вариантов выполнения, за исключением контроля входных параметров на тип данных. Программа из расширенной части создает рисунок идентичный рисунку из варианта задания.