Федеральное государственное образовательное бюджетное учреждение высшего образования

«Финансовый университет при правительстве Российской Федерации»

(Финансовый университет)

Колледж информатики и программирования

МДК.05.02 Разработка кода информационных систем

Группа: 3ИСИП-723

Лабораторная работа 1

На тему: Основы JavaScript. Переменные, типы данных, операции

Выполнил:

Е.Д. Сидоров

Принял:

Р.Р. Абзалимов

Москва

2026

Содержание

ВВЕДЕНИЕ ..... **Error! Bookmark not defined.**
Практическая часть 4
Шаг 2: Работа с переменными 4
Шаг 3: Изучение типов данных 4
Шаг 4: Математические операции 5
Шаг 5: Работа со строками 6
Шаг 6: Диалоговые окна и вывод 7
Консоль 9
Контрольные вопросы 10

ВВЕДЕНИЕ

Цель: изучить базовый синтаксис JavaScript, научиться работать с переменными, различными типами данных и основными операциями.

Задачи:

1. Освоить подключение JavaScript к HTML-документу
2. Изучить объявление переменных (let, const)
3. Познакомиться с типами данных: number, string, boolean, null, undefined, object, symbol
4. Освоить математические операции и приоритет операций
5. Научиться работать со строками: шаблонные строки, определение длины 6. Практически использовать: alert(), console.log(), typeof, parseInt(), parseFloat()

Теоретическая справка:

JavaScript — язык программирования для создания интерактивных вебстраниц. Переменные хранят данные, которые могут изменяться в ходе выполнения программы. Типы данных определяют характер информации, хранящейся в переменных. Операции позволяют манипулировать данными.

Оборудование и программное обеспечение:

- Компьютер с доступом в интернет
- Браузер (Chrome, Firefox, Edge)
- Текстовый редактор (VS Code, Sublime Text, Notepad++)
- Консоль разработчика в браузере

Практическая часть

Шаг 2: Работа с переменными

1. Объявите переменные с использованием `let` и `const`

2. Присвойте переменным различные значения

3. Продемонстрируйте разницу между `let` (изменяемая переменная) и `const` (неизменяемая константа)

4. Выведите значения переменных в консоль с помощью `console.log ()

```
// Объявление переменных использованием let и const
let изменяемаяПеременная = 10;
const неизменяемаяКонстанта=20;
```

Шаг 3: Изучение типов данных

1. Создайте переменные разных типов данных: number, string, boolean

2. Создайте переменные со значениями null и undefined

3. Создайте простой объект с несколькими свойствами

4. Используйте оператор `typeof` для определения типа каждой переменной 5. Выведите результаты в консоль

```
// Создание переменных разных типов данных
const число = 42;
   const строка = "Привет, мир!";
   const булево = true;
const пустое = null;
   const неопределенное = undefined;
   const объект = {
имя: "Иван",
возраст: 25,
      студент: true
   };
   const массив = [1, 2, 3, 4, 5];
   const функция = function() { return "Я функция"; };
```

Шаг 4: Математические операции

1. Создайте переменные с числовыми значениями
2. Выполните основные арифметические операции: сложение, вычитание, умножение, деление
3. Используйте операции инкремента (++) и декремента (--)
4. Продемонстрируйте приоритет операций с помощью скобок
5. Используйте оператор остатка от деления (%)

```
// Основные арифметические операции
console.log(`${a} + ${b} = ${a + b}`);
console.log(`${a} - ${b} = ${a - b}`);     console.log(`${a}
* ${b} = ${a * b}`);     console.log(`${a} / ${b} = ${a /
b}`);

   // Операции инкремента и декремента
console.log("c до инкремента:", c);     c++;
console.log("c после инкремента (c++):", c);
c--;
   console.log("c после декремента (c--):", c);

   // Префиксный и постфиксный инкремент
let x = 5;
   let y = x++; // Сначала присваивание, потом увеличение
console.log(`Постфиксный инкремент: x=${x}, y=${y}`);

x         = 5;
y         = ++x; // Сначала увеличение, потом присваивание     console.log(`Префиксный
инкремент: x=${x}, y=${y}`);

   // Приоритет операций     const результат1 = 2 + 3 * 4; // 14
(умножение выполняется первым)     const результат2 = (2 + 3) * 4; //
20 (скобки меняют приоритет)
   console.log(`2 + 3 * 4 = ${результат1} (умножение имеет более высокий приоритет)`);
console.log(`(2 + 3) * 4 = ${результат2} (скобки меняют приоритет)`);

   // Оператор остатка от деления
   console.log(`${a} % ${b} = ${a % b} (остаток от деления)`);
   console.log(`Проверка на четность: 10 % 2 = ${10 % 2}, 11 % 2 = ${11 % 2}`);

   // Другие математические операции
   console.log(`Возведение в степень: 2 ** 3 = ${2 ** 3}`);
console.log(`Унарный минус: -${a} = ${-a}`);
```

5: Работа со строками

1. Создайте строковые переменные (одинарные и двойные кавычки)

2. Используйте шаблонные строки (обратные кавычки) для вставки переменных

3. Определите длину строк с помощью свойства `length`

4. Продемонстрируйте конкатенацию строк

5. Используйте `parseInt()` и `parseFloat()` для преобразования строк в числа

```
// Создание строковых переменных
   const строка1 = 'Строка в одинарных кавычках';
const строка2 = "Строка в двойных кавычках";     const
имя = "Анна";
   const возраст = 25;

   // Шаблонные строки (обратные кавычки)
   const приветствие = `Привет, меня зовут ${имя} и мне ${возраст} лет!`;
console.log("Шаблонная строка:", приветствие);

   // Многострочные строки     const
многострочнаяСтрока = `Это
многострочная     строка
   с использованием     шаблонных
строк`;     console.log("Многострочная
строка:");
console.log(многострочнаяСтрока);

   // Определение длины строк     console.log(`Длина строки "${строка1}":
${строка1.length} символов`);     console.log(`Длина строки "${строка2}":
${строка2.length} символов`);     console.log(`Длина шаблонной строки:
${приветствие.length} символов`);

   // Конкатенация строк
   const конкатенация1 = строка1 + ". " + строка2;
   const конкатенация2 = "Имя: " + имя + ", Возраст: " + возраст;
console.log("Конкатенация строк (оператор +):", конкатенация1);
console.log("Конкатенация с переменными:", конкатенация2);

   // Преобразование строк в числа
const строкаЧисло1 = "42";
const строкаЧисло2 = "3.14";
   const строкаНеЧисло = "42abc";
```

6: Диалоговые окна и вывод

1. Используйте `alert()` для вывода информации во всплывающем окне

2. Используйте `console.log ()` для вывода отладочной информации

3. Сравните эти два метода вывода информации

```
    // Использование console.log() для вывода отладочной информации
console.log("=== Шаг 6: Диалоговые окна и вывод ===");     console.log("Это
сообщение видно только в консоли разработчика.");
console.log("console.log() не прерывает выполнение программы.");

    // Различные методы console
    console.info("Информационное сообщение");
console.warn("Предупреждение");
    console.error("Сообщение об ошибке");

    // Группировка сообщений
console.group("Группа сообщений");
console.log("Сообщение 1 внутри группы");
console.log("Сообщение 2 внутри группы");
console.groupEnd();

    // Вывод таблицы
const студенты = [
        { имя: "Иван", возраст: 20, курс: 2 },
        { имя: "Мария", возраст: 21, курс: 3 },
        { имя: "Алексей", возраст: 19, курс: 1 }
    ];
    console.table(студенты);

    // Вывод объектов
const автомобиль = {
марка: "Toyota",
модель: "Camry",
год: 2020,
        цвет: "серебристый"
    };
    console.dir(автомобиль);

    // Выводим результаты на страницу
output.innerHTML = `
    Сообщения были выведены в консоль.

    Отличия alert() и console.log():
1. alert() блокирует выполнение скрипта, console.log() - нет
2. alert() показывает окно пользователю, console.log() - только разработчику
3. alert() используется для важных сообщений, console.log() - для отладки
```

Консоль



Рисунок 1 Консоль (1часть)



Рисунок 2 Консоль (2часть)

Контрольные вопросы:

1. `let`, `const`, `var`:

   - `var` - устаревшее, работает во всей функции

   - `let` - можно менять, работает только в блоке {} - `const` - нельзя менять, работает только в блоке {}

2. Примитивные типы:

   Строка, число, логическое, null, undefined, символ, большое число

3. Определение типа:

   Оператором `typeof`

4. `typeof null`:**

   Возвращает `"object"` — это ошибка языка

5. `==` vs `===`:

   `==` сравнивает с приведением типов, `===` сравнивает строго (тип + значение)

6. Преобразование строки в число:

   `parseInt()` для целых, `parseFloat()` для дробных

7. Шаблонные строки:

   Текст в обратных кавычках `` `текст ${переменная}` ``

8. Длина строки:

   Свойство `.length`

9. Приоритет операций:

   Сначала скобки, потом умножение/деление, потом сложение/вычитание

10. `alert()` vs `console.log()`:

    `alert()` — для пользователя (блокирует работу)

    `console.log()` — для отладки (не блокирует)

Самостоятельная работа

Задание 1: Расчет ИМТ

```
// Расчет ИМТ
        const bmi = weight / (height * height);

        // Определение
категории           let category = '';
let color = '';

        if (bmi < 18.5) {
            category = 'Недостаточный вес';
color = '#2196F3'; // синий
        } else if (bmi < 25) {
            category = 'Нормальный вес';
color = '#4CAF50'; // зеленый
        } else if (bmi < 30) {
            category = 'Избыточный вес';
            color = '#FF9800'; // оранжевый
        } else {
            category     =     'Ожирение';
color = '#F44336'; // красный
        }
```

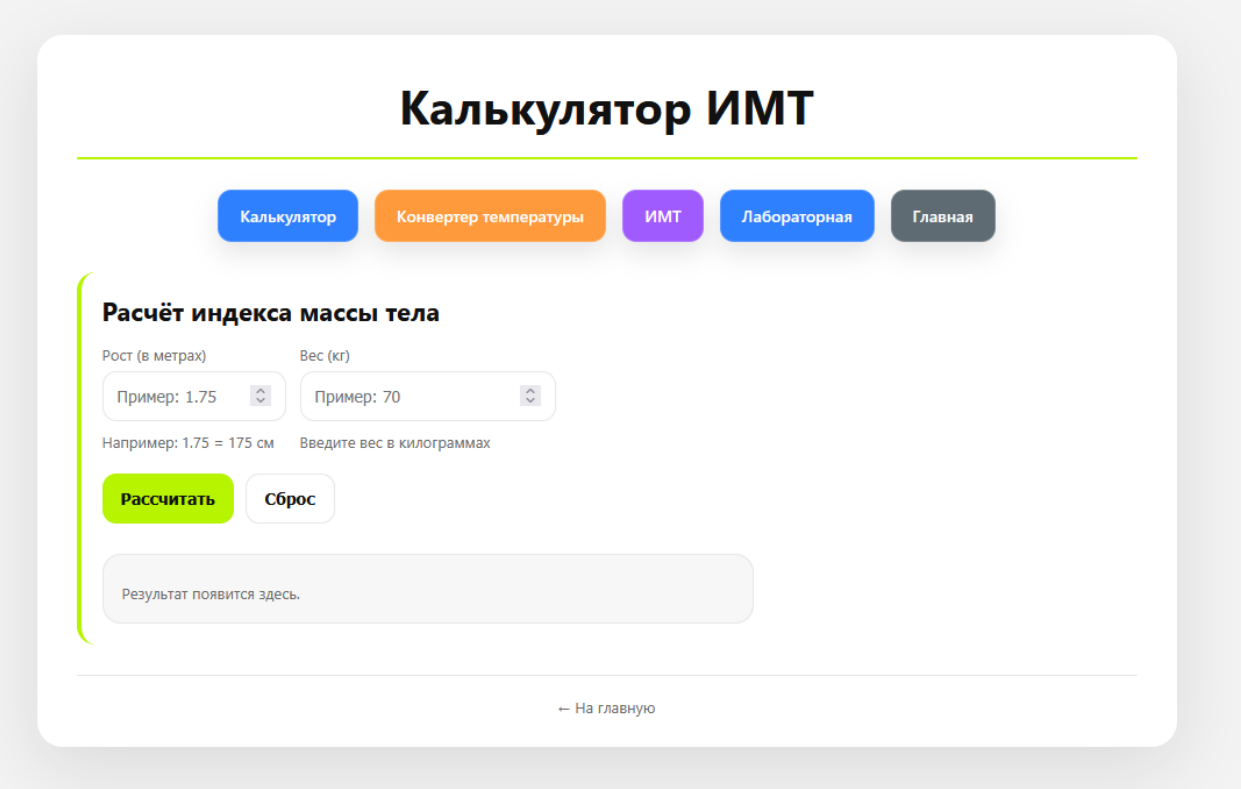


Рисунок 3 ИМТ

Задание 2: Калькулятор

```
// Выполнение операции
```

```
        switch(operator) {
case '+':
                result = num1 + num2;
                operation = `${num1} + ${num2}`;
break;              case '-':
                result = num1 - num2;
                operation = `${num1} - ${num2}`;
break;              case '*':                   result =
num1 * num2;
                operation = `${num1} × ${num2}`;
break;              case '/':                   if (num2
=== 0) {
                   resultDiv.innerHTML = '<span class="error">Ошибка: деление на ноль!</span>';
return;
                }
                result = num1 / num2;
                operation = `${num1} ÷
${num2}`;                   break;              case
'%':                result = num1 % num2;
                operation = `Остаток от ${num1} ÷ ${num2}`;
break;              case '**':
                result = Math.pow(num1, num2);
                operation =
`${num1}<sup>${num2}</sup>`;                    break;
default:
                resultDiv.innerHTML = '<span class="error">Неизвестная операция</span>';
return;
        }
```

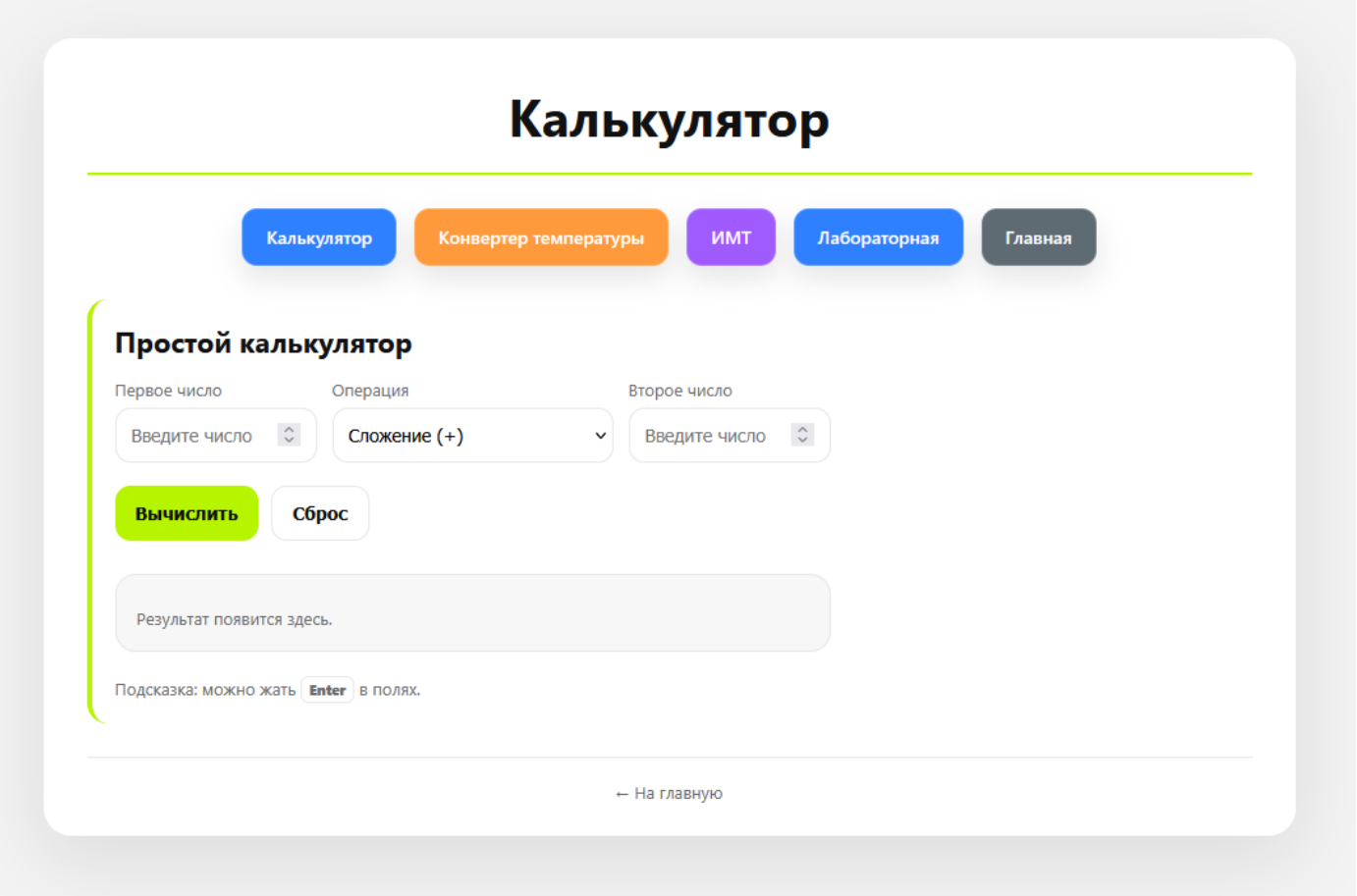


Рисунок 4 Калькулятор

Задание 3: Цельсия-Фаренгейты

```
// Цельсий → Фаренгейт: °F = (°C × 9/5) + 32 convertedTemp = (temperature * 9/5) + 32; formula = `°F = (${temperature} × 9/5) + 32 = ${convertedTemp.toFixed(2)}`; title = `Результат: ${temperature}°C = ${convertedTemp.toFixed(2)}°F`; originalText = `Исходное значение: ${temperature}°C`;
        } else {
            // Фаренгейт → Цельсий: °C = (°F - 32) × 5/9
convertedTemp = (temperature - 32) * 5/9;
            formula = `°C = (${temperature} - 32) × 5/9 = ${convertedTemp.toFixed(2)}`;
title = `Результат: ${temperature}°F = ${convertedTemp.toFixed(2)}°C`;
originalText = `Исходное значение: ${temperature}°F`;
        }
```

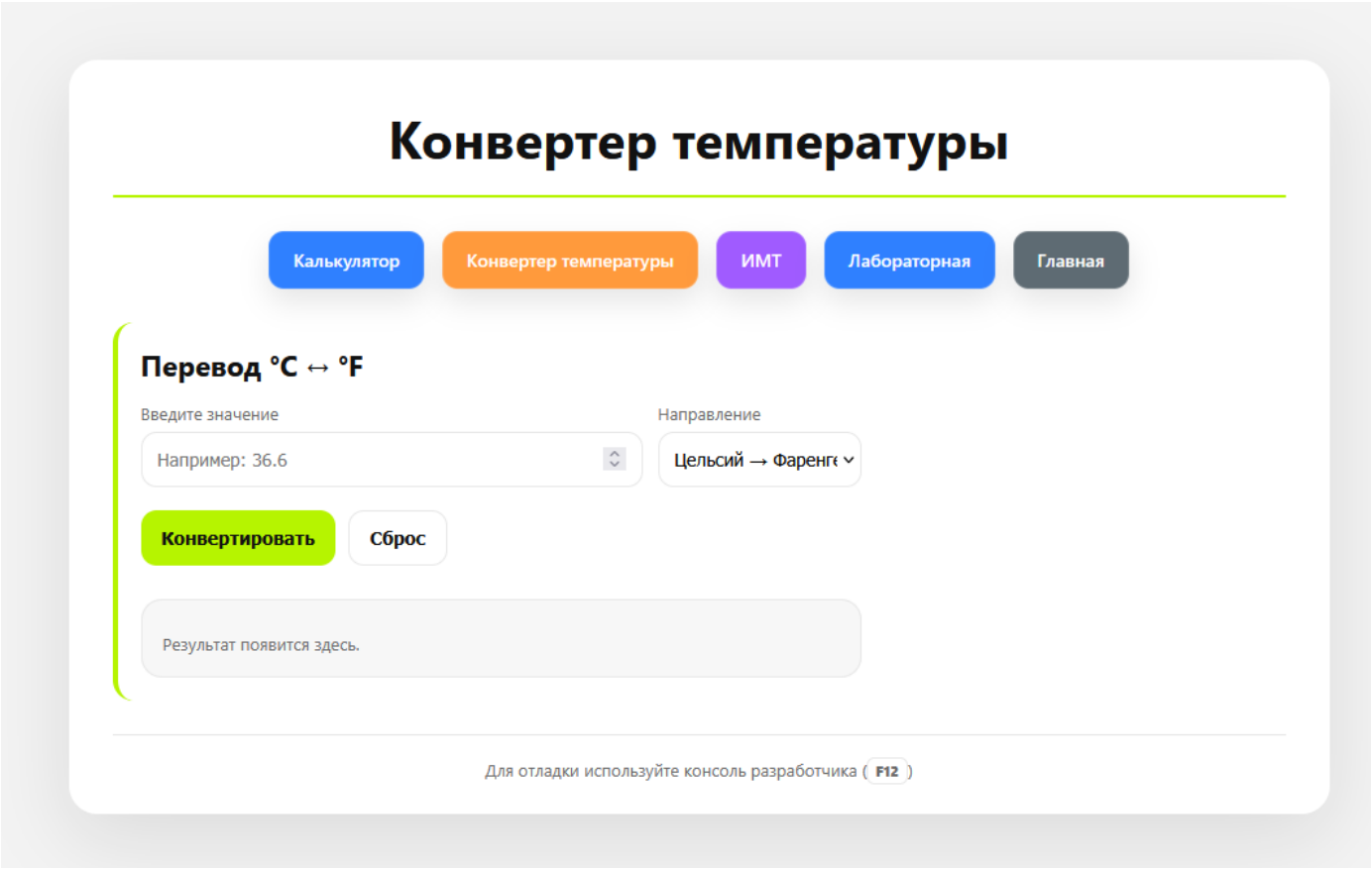


Рисунок 5 Температура

Заключение

В ходе выполнения лабораторной работы были успешно изучены и применены на практике основные принципы программирования на JavaScript. Была создана структура проекта с использованием внешнего подключения файлов, что улучшает организацию кода и его повторное использование. В процессе работы освоено объявление переменных с помощью let и const, продемонстрирована разница между изменяемыми и неизменяемыми значениями. Изучены различные типы данных в JavaScript, включая примитивные типы и объекты, а также работа с оператором typeof для проверки типов. Практически выполнены математические операции с учетом приоритета, включая сложение, вычитание, умножение и деление. Освоена работа со строками, включая использование шаблонных строк для удобной вставки переменных и определение длины строк. На практике применены методы вывода информации через alert() и console.log(), что позволило понять их различия и оптимальные сценарии использования. В результате работы сформированы базовые навыки JavaScript-программирования, которые являются фундаментом для дальнейшего изучения веб-разработки и создания интерактивных приложений. Полученные знания позволяют уверенно работать с переменными, типами данных, операциями и отладкой кода.