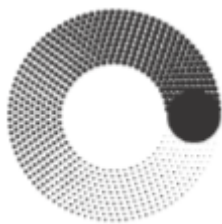


**федеральное государственное автономное образовательное
учреждение высшего образования**



МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет информационных технологий

Кафедра Информатики и информационных технологий

направление подготовки

09.03.02 «Информационные системы и технологии»

ЛАБОРАТОРНАЯ РАБОТА № 4


Дисциплина: Функциональное программирование _____

Тема: Применение функционального программирования в TypeScript

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

Грохотова Е.Д
(Фамилия И.О.)

Дата, подпись 27.04.2025
(Дата)


(Подпись)

Проверил: _____
(Фамилия И.О., степень, звание) (Оценка)

Дата, подпись _____
(Дата) (Подпись)

Замечания: _____

Москва 2024

Цель: Применить принципы функционального программирования для разработки небольшого веб-приложения на TypeScript.

Задание:

Разработайте веб-приложение "Калькулятор", которое позволяет пользователю выполнять следующие операции:

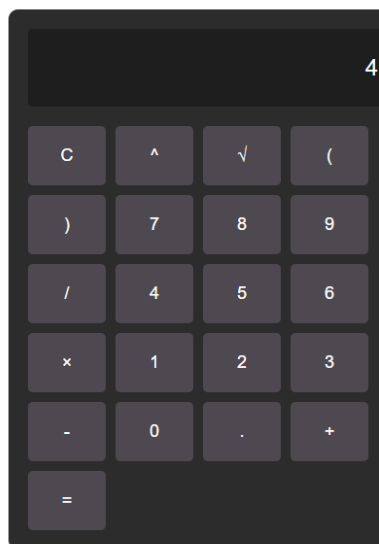
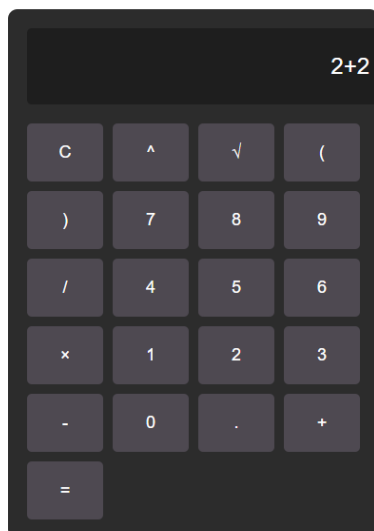
- Сложение, вычитание, умножение и деление.
- Возведение в степень.
- Вычисление квадратного корня.

Требования:

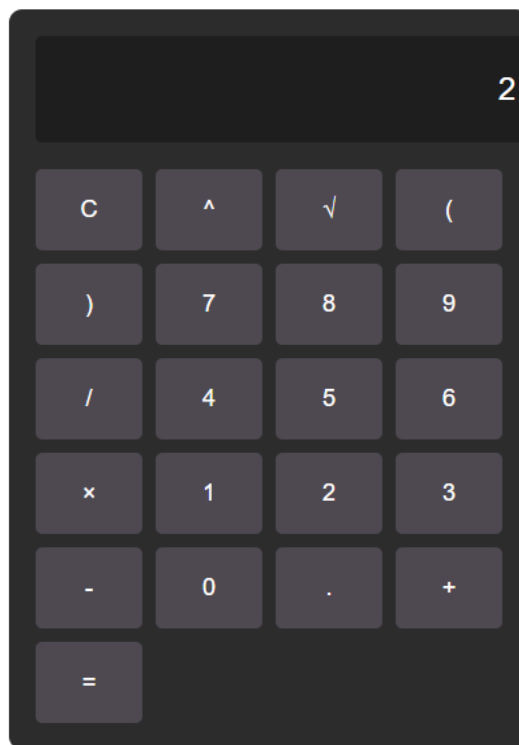
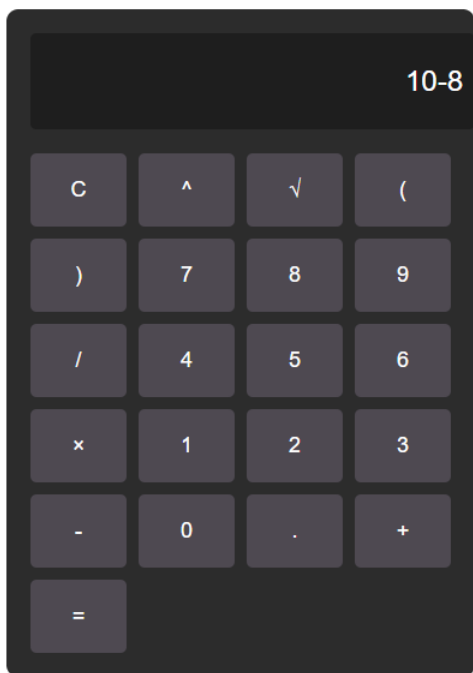
- Используйте принципы функционального программирования, такие как иммутабельность данных и чистые функции.
- Используйте функции высшего порядка для обработки данных и создания новых функций.
- Веб-приложение должно быть реализовано с использованием HTML, CSS и TypeScript.
- Интерфейс должен быть интуитивно понятным и удобным для пользователя.

Ссылка на гит: <https://github.com/Katyadra/FuncProgramming4>

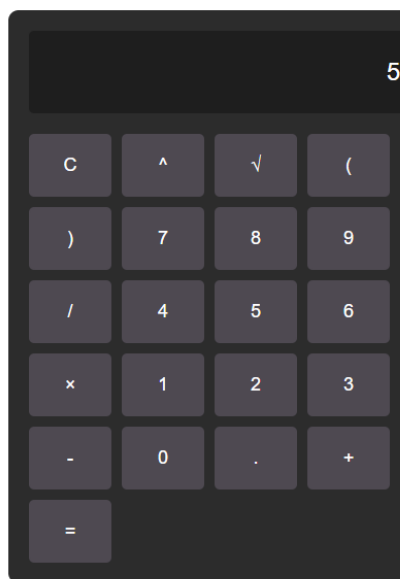
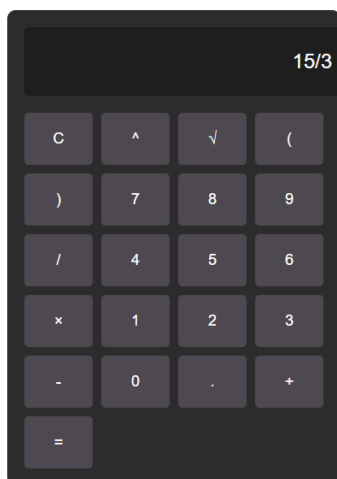
Сложение:



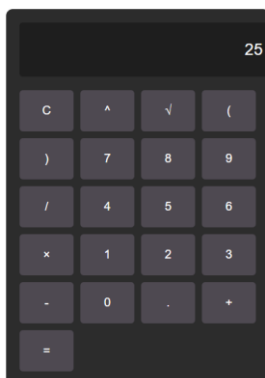
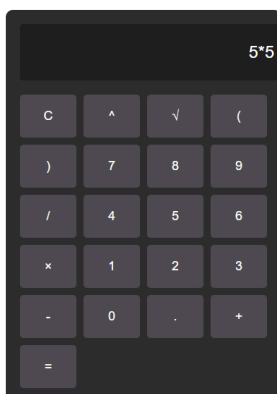
Вычитание:



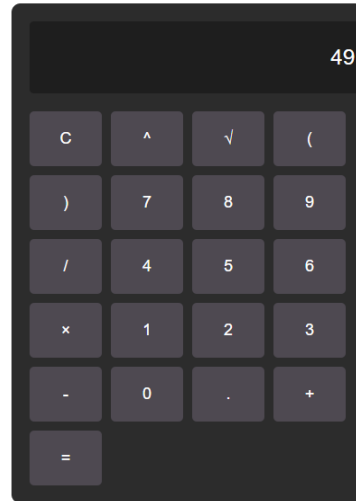
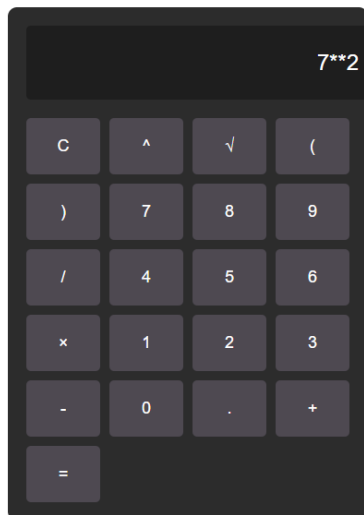
Деление:



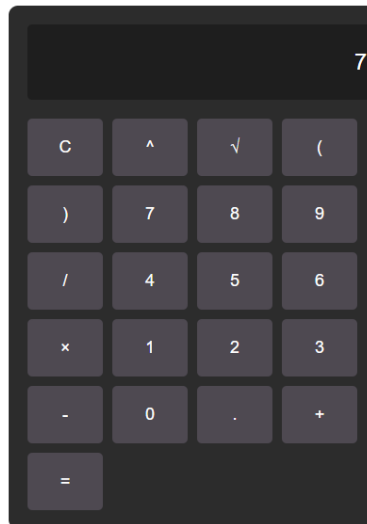
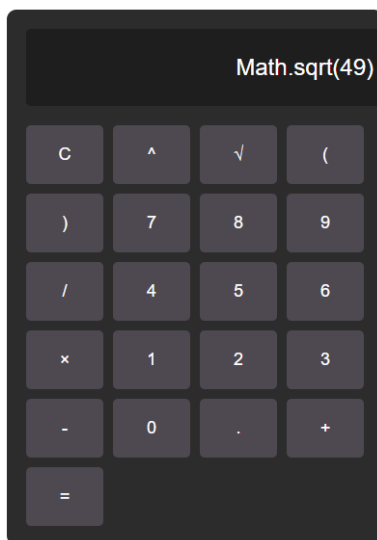
Умножение:



Возведение в степень:



Квадратный корень:



Листинг кода:

App.ts

```
type CalculatorState = { display: string };

// Начальное состояние
const initialState: CalculatorState = { display: "" };

// Чистые функции для обновления состояния
const appendToDisplay = (state: CalculatorState, value: string): CalculatorState => {
  {
    if (value === "√") {
      // Добавляем "Math.sqrt(" вместо "Math.sqrt()"
      return { display: state.display + "Math.sqrt(" };
    }
    return { display: state.display + value };
  }
};
```

```

const clearDisplay = (): CalculatorState => initialState;

// Проверка баланса скобок
const isBalanced = (expr: string): boolean => {
  const stack: string[] = [];
  for (const char of expr) {
    if (char === "(") stack.push("(");
    if (char === ")") {
      if (stack.length === 0) return false;
      stack.pop();
    }
  }
  return stack.length === 0;
};

// Функция вычисления результата
const computeResult = (state: CalculatorState): CalculatorState => {
  try {
    if (!isBalanced(state.display)) {
      throw new Error("Незакрытые скобки");
    }
    const result = eval(state.display);
    return { display: Number.isNaN(result) ? "Ошибка" : result.toString() };
  } catch (error) {
    return { display: "Ошибка" };
  }
};

// Инициализация приложения
document.addEventListener("DOMContentLoaded", () => {
  let state: CalculatorState = initialState;
  const display = document.getElementById("display") as HTMLInputElement;

  // Обновление интерфейса
  const render = (state: CalculatorState) => {
    display.value = state.display;
  };

  // Обработчики событий
  document.querySelectorAll("button").forEach(button => {
    button.addEventListener("click", () => {
      const action = button.getAttribute("data-action");
      if (!action) return;

      state = action === "clear"
        ? clearDisplay()
        : action === "="
          ? computeResult(state)
          : appendToDisplay(state, action);

      render(state);
    });
  });
});

```

```
    });  
  });  
});
```

Index.css

```
#calculator {  
  width: 350px;  
  margin: 20px auto;  
  padding: 20px;  
  background: #2c2c2c;  
  border-radius: 10px;  
}  
  
#display {  
  width: 100%;  
  height: 60px;  
  margin-bottom: 20px;  
  font-size: 24px;  
  text-align: right;  
  padding: 10px;  
  background: #1e1e1e;  
  color: white;  
  border: none;  
  border-radius: 5px;  
}  
  
#buttons {  
  display: grid;  
  grid-template-columns: repeat(4, 1fr);  
  gap: 10px;  
}  
  
button {  
  padding: 20px;  
  font-size: 18px;  
  border: none;  
  border-radius: 5px;  
  background: #4e4951;  
  color: white;  
  cursor: pointer;  
}  
  
button:hover {  
  background: #37323a;  
}
```

Index.html

```
<!DOCTYPE html>  
<html lang="ru">
```

```

<head>
  <meta charset="UTF-8">
  <title>Калькулятор</title>
  <link rel="stylesheet" href="index.css">
</head>
<body>
  <div id="calculator">
    <input type="text" id="display" disabled>
    <div id="buttons">
      <button data-action="clear">C</button>
      <button data-action="**">^</button>
      <button data-action="√">√</button>
      <button data-action="(">(</button>
      <button data-action=")">)</button>
      <button data-action="7">7</button>
      <button data-action="8">8</button>
      <button data-action="9">9</button>
      <button data-action="/">/</button>
      <button data-action="4">4</button>
      <button data-action="5">5</button>
      <button data-action="6">6</button>
      <button data-action="*">×</button>
      <button data-action="1">1</button>
      <button data-action="2">2</button>
      <button data-action="3">3</button>
      <button data-action="-">-</button>
      <button data-action="0">0</button>
      <button data-action=".">.</button>
      <button data-action="+">+</button>
      <button data-action="=">=</button>
    </div>
  </div>
  <script src="dist/app.js"></script>
</body>
</html>

```

Tsconfig.json

```

{
  "compilerOptions": {
    "target": "ES6",
    "module": "ES6",
    "outDir": "./dist",
    "strict": true,
    "lib": ["DOM", "ES6"]
  }
}

```