кафедра

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

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

ОТЧЕТ О ПРАКТИЧЕСКОМ ЗАДАНИИ №3-4

«Язык TypeScript, библиотеки и фреймворки JavaScript»
по курсу: ИТ-модуль «JavaScript, его библиотеки и фреймворки в Frontend-разработке»

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № _____ _____ _____
подпись, дата | инициалы, фамилия

Санкт-Петербург 2024

Цель работы:

Формирование практических навыков работы с прототипами и классами в JavaScript.

Задание:

1. Создайте прототип "Фигура", который будет иметь методы для вычисления площади и периметра фигуры. Напишите классы "Прямоугольник" и "Круг", которые наследуются от прототипа "Фигура" и реализуют соответствующие методы.

2. Создайте класс "Студент", который будет иметь защищенные свойства "имя", "возраст" и "средний балл". Напишите методы для изменения и получения значений этих свойств.

3. Создайте класс "Калькулятор", который будет иметь методы для выполнения основных математических операций (сложение, вычитание, умножение, деление).

Выполнение задания:

1 пункт:

```
function Figure() {}

Figure.prototype.area = function() {
  throw new Error("This method should be implemented in subclass");
};

Figure.prototype.perimeter = function() {
  throw new Error("This method should be implemented in subclass");
};
```

Рисунок 1

На рисунке 1 создание прототипа «Фигура».

```
function Rectangle(width, height) {
  Figure.call(this);
  this.width = width;
  this.height = height;
}

Rectangle.prototype = Object.create(Figure.prototype);
Rectangle.prototype.constructor = Rectangle;

Rectangle.prototype.area = function() {
  return this.width * this.height;
};

Rectangle.prototype.perimeter = function() {
  return 2 * (this.width + this.height);
};
```

Рисунок 2

На рисунке 2 создание класса «Прямоугольник».

```
function Circle(radius) {
  Figure.call(this);
  this.radius = radius;
}

Circle.prototype = Object.create(Figure.prototype);
Circle.prototype.constructor = Circle;

Circle.prototype.area = function() {
  return Math.PI * this.radius * this.radius;
};

Circle.prototype.perimeter = function() {
  return 2 * Math.PI * this.radius;
};
```

Рисунок 3

На рисунке 3 создание класса «Круг».

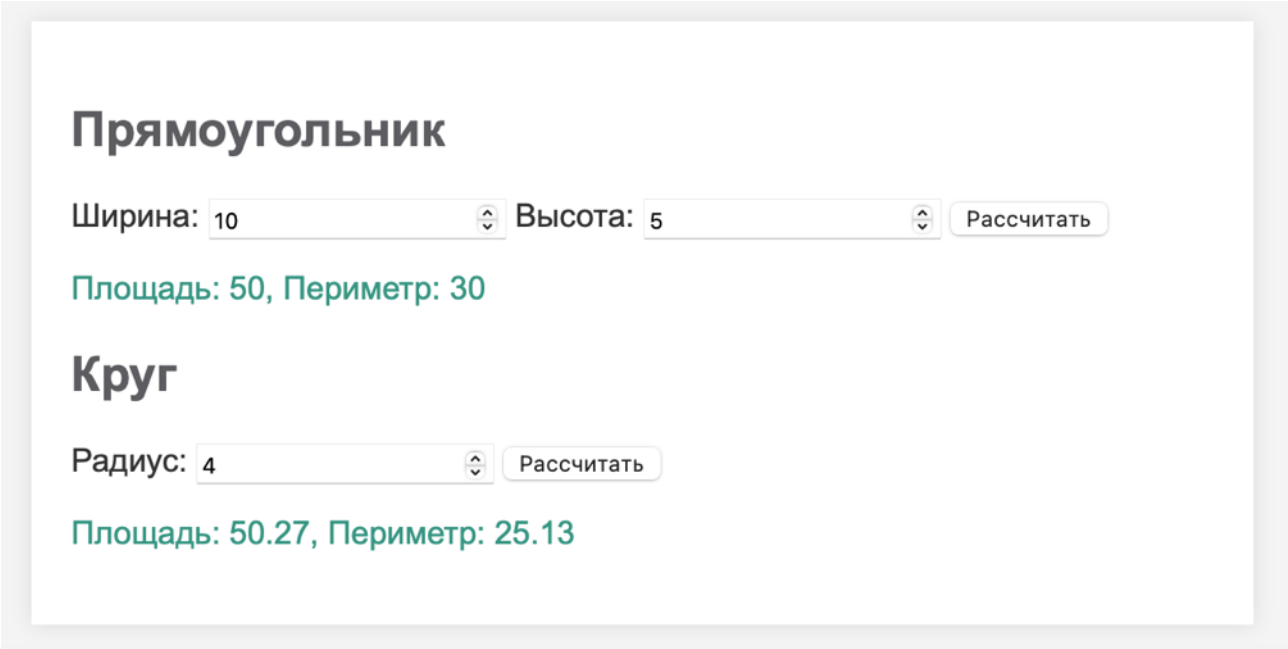

Рисунок 4

На рисунке 4 пример тестирования.

Для прямоугольника с размерами 10x5:
Площадь: 10 * 5 = 50
Периметр: 2 * (10 + 5) = 30
Для круга с радиусом 4:
Площадь: π * 4^2 ≈ 50.27
Периметр (длина окружности): 2 * π * 4 ≈ 25.13

На рисунке 4 видно те же значения, которые мы считали вручную. Таким образом можно сделать вывод, что классы успешно реализованы и протестированы.

2 пункт:

```
class Student {
  #name;
  #age;
  #grade;

  constructor(name, age, grade) {
    this.#name = name;
    this.#age = age;
    this.#grade = grade;
  }

  getName() {
    return this.#name;
  }

  setName(name) {
    this.#name = name;
  }

  getAge() {
    return this.#age;
  }

  setAge(age) {
    this.#age = age;
  }

  getGrade() {
    return this.#grade;
  }

  setGrade(grade) {
    this.#grade = grade;
  }
}
```

Рисунок 5

На рисунке 5 написание класса Student.
#name, #age, #grade — это закрытые свойства. Они не доступны за пределами класса, что делает их защищёнными.
constructor — это специальный метод для создания и инициализации объектов, созданных с помощью class.
getName, setName, getAge, setAge, getGrade, setGrade — это методы для доступа и изменения свойств.

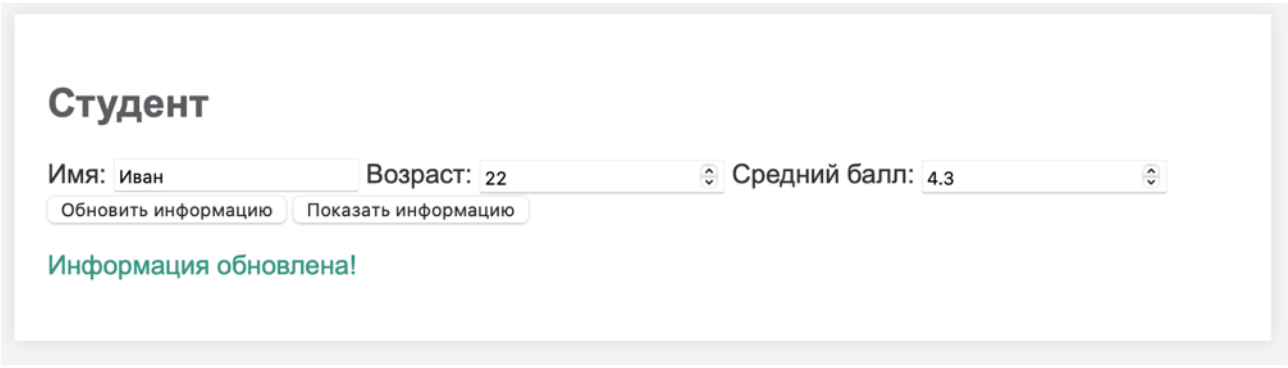

Рисунок 6

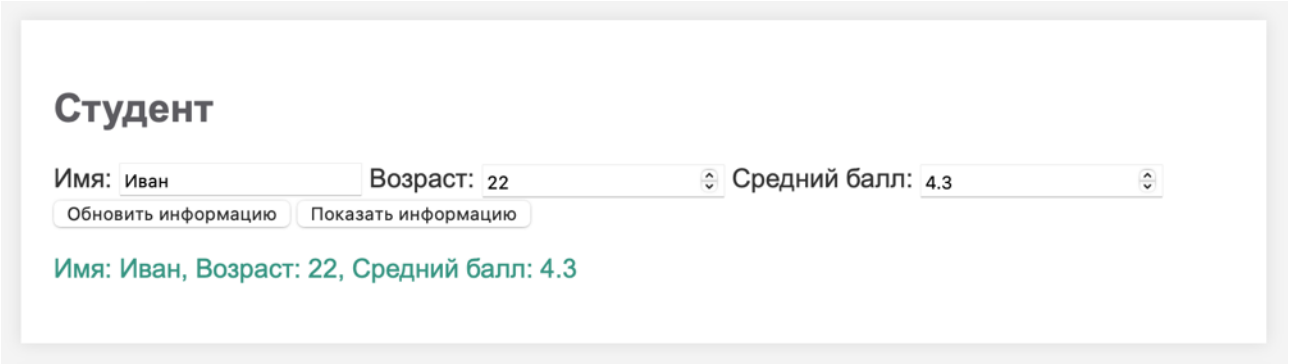

Рисунок 7

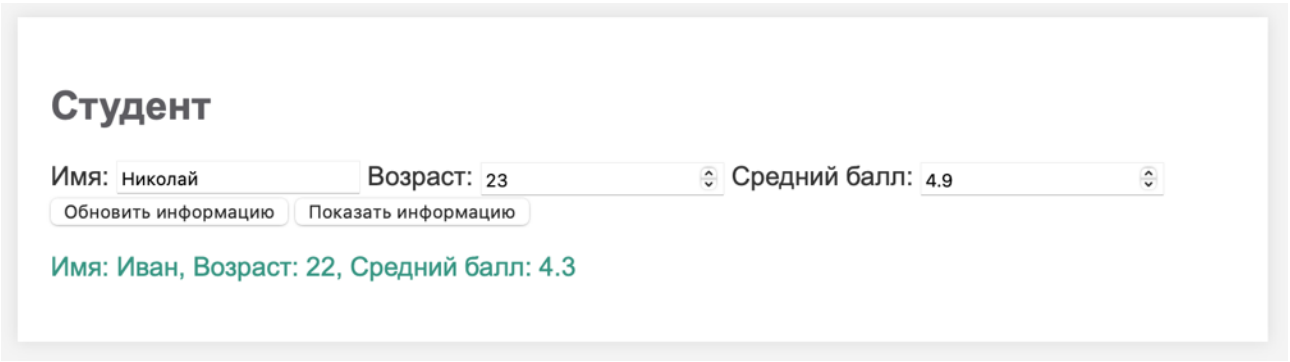

Рисунок 8

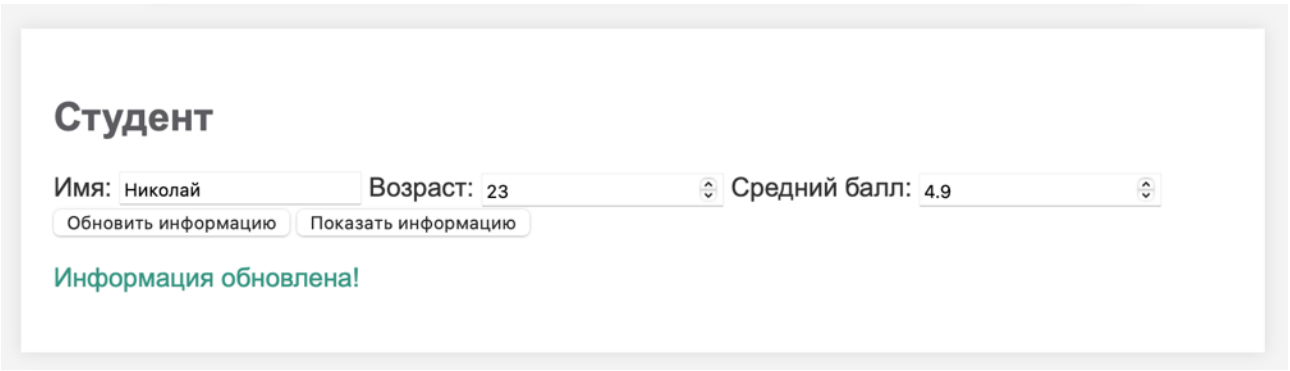

Рисунок 9

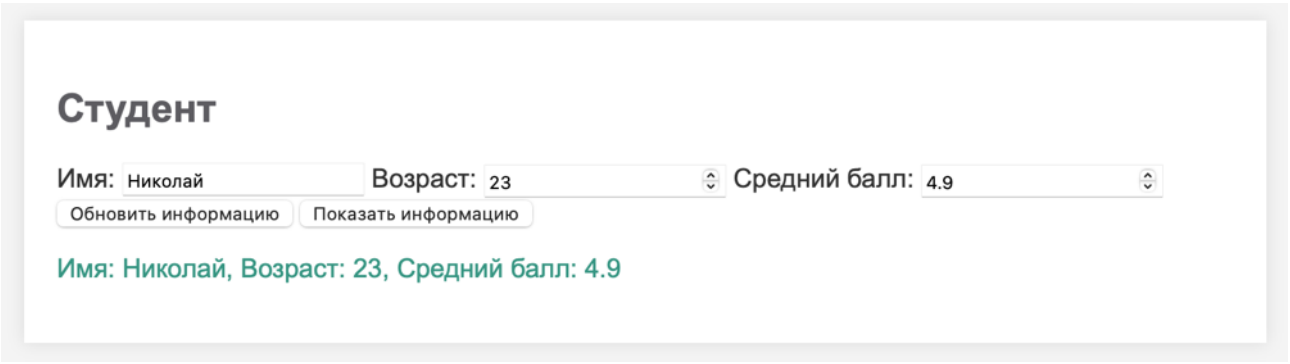

Рисунок 10

На рисунках 6-10 пример тестирования.
Мы создали экземпляр класса Student, проверили начальные значения свойств, изменили их с помощью методов setName, setAge, setGrade и затем проверили обновленные значения. Это подтверждает, что методы для изменения и получения значений работают корректно.

3 пункт:

```
class Calculator {
  add(a, b) {
    return a + b;
  }

  subtract(a, b) {
    return a - b;
  }

  multiply(a, b) {
    return a * b;
  }

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

Рисунок 11

На рисунке 9 определяем класс Calculator.

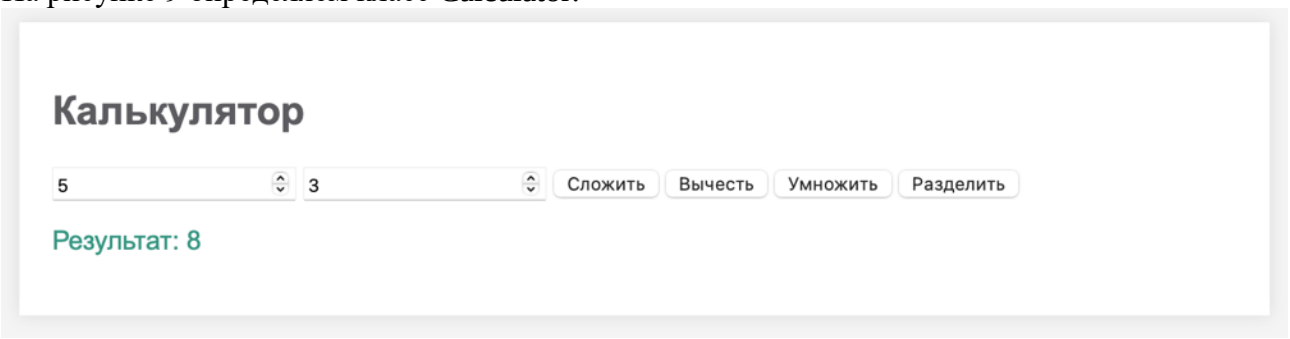

Рисунок 12

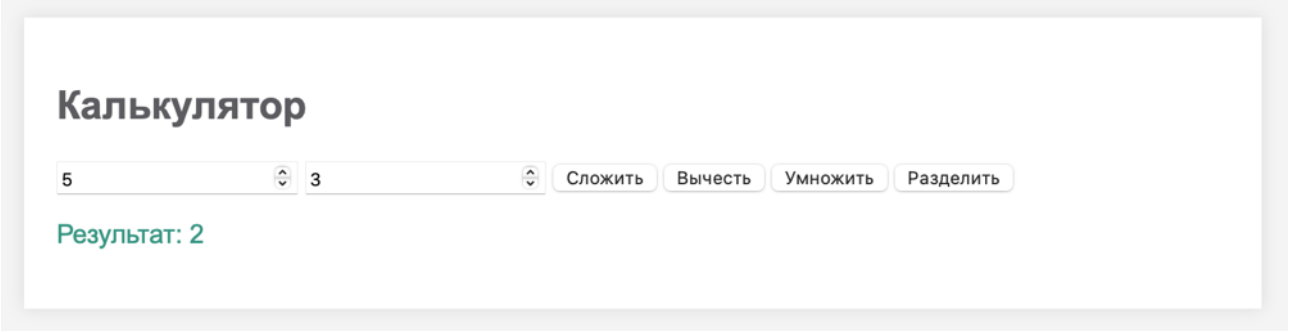

Рисунок 13

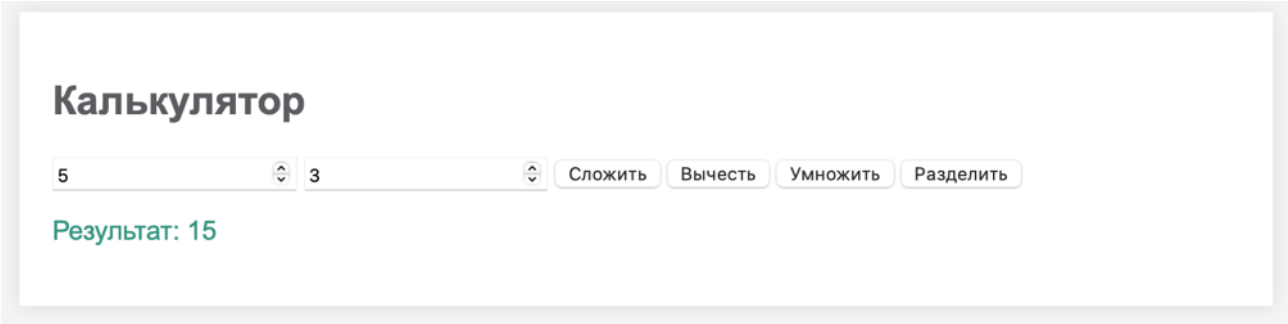

Рисунок 14

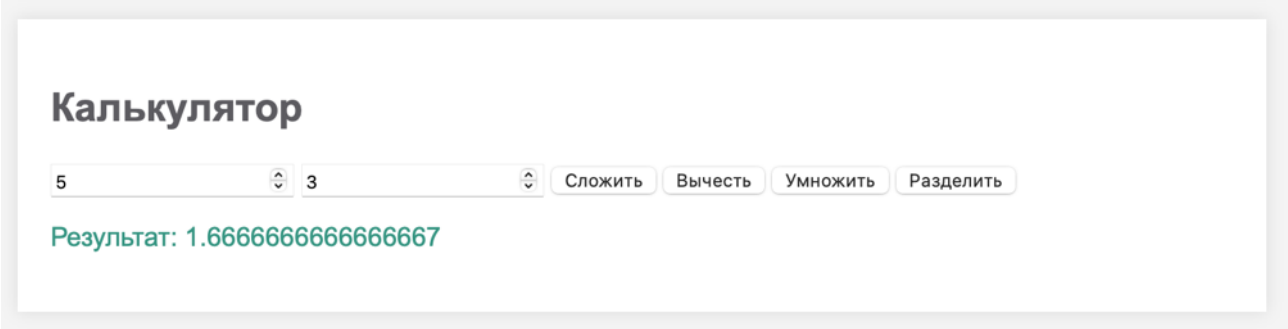

Рисунок 15

На рисунках 12 -15 пример тестирования.
Сложение 5+3 дало 8,
Вычитание 5−3 дало 2,
Умножение 5×3 дало 15,
Деление 5÷3 дало примерно 1.67.

На рисунках видно те же значения, которые мы считали вручную.
Это означает, что класс Calculator работает правильно и выполняет основные математические операции.

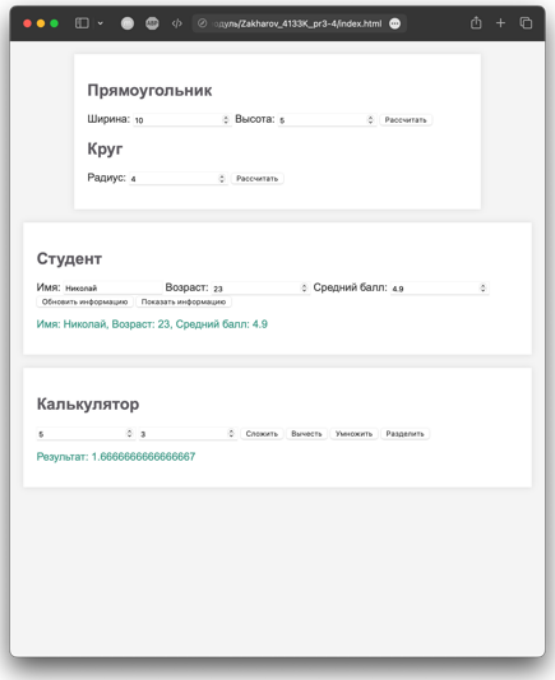

Рисунок 16 – общий вид страницы

Коды веб-страниц:

index.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Задание 3-4</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 20px;
            color: #333;
        }
        .container {
            width: 80%;
            margin: 0 auto;
        }
        .block {
            background-color: white;
            padding: 20px;
            margin-bottom: 20px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }
        h2 {
```

```
            color: #5D5C61;
        }
        p {
            color: #379683;
        }
    </style>
</head>
<body>
    <div class="container">
        <div class="block">
            <h2>Прямоугольник</h2>
            <label for="rectangle-width">Ширина:</label>
            <input type="number" id="rectangle-width" value="10">
            <label for="rectangle-height">Высота:</label>
            <input type="number" id="rectangle-height" value="5">
            <button onclick="calculateRectangle()">Рассчитать</button>
            <p id="rectangle-result"></p>
            <h2>Круг</h2>
            <label for="circle-radius">Радиус:</label>
            <input type="number" id="circle-radius" value="4">
            <button onclick="calculateCircle()">Рассчитать</button>
            <p id="circle-result"></p>
        </div>
    </div>
    <div class="block">
      <h2>Студент</h2>
```

```
    <label for="student-name">Имя:</label>
    <input type="text" id="student-name" placeholder="Имя">
    <label for="student-age">Возраст:</label>
    <input type="number" id="student-age" placeholder="Возраст">
    <label for="student-grade">Средний балл:</label>
    <input type="number" id="student-grade" placeholder="Средний балл" step="0.1">
    <button onclick="updateStudentInfo()">Обновить информацию</button>
    <button onclick="showStudentInfo()">Показать информацию</button>
    <p id="student-info"></p>
  </div>
  </div>
    <div class="block">
      <h2>Калькулятор</h2>
      <input type="number" id="calc-num1">
      <input type="number" id="calc-num2">
      <button onclick="calculate('add')">Сложить</button>
      <button onclick="calculate('subtract')">Вычесть</button>
      <button onclick="calculate('multiply')">Умножить</button>
      <button onclick="calculate('divide')">Разделить</button>
      <p id="calculator-result"></p>
    </div>
  </div>

    <script src="figures.js"></script>
    <script src="student.js"></script>
    <script src="calculator.js"></script>
```

```
<script>
    function calculateRectangle() {
        var width = parseFloat(document.getElementById("rectangle-width").value);
        var height = parseFloat(document.getElementById("rectangle-height").value);
        var rect = new Rectangle(width, height);
        document.getElementById("rectangle-result").innerHTML =
            `Площадь: ${rect.area()}, Периметр: ${rect.perimeter()}`;
    }
    function calculateCircle() {
        var radius = document.getElementById("circle-radius").value;
        var circle = new Circle(radius);
        document.getElementById("circle-result").innerHTML =
            `Площадь: ${circle.area().toFixed(2)}, Периметр: ${circle.perimeter().toFixed(2)}`;
    }




    var student;

    function updateStudentInfo() {
        var name = document.getElementById("student-name").value;
        var age = document.getElementById("student-age").value;
        var grade = document.getElementById("student-grade").value;
        student = new Student(name, age, grade);
```

```
        document.getElementById("student-info").innerHTML = "Информация обновлена!";
    }

    function showStudentInfo() {
        if(student) {
            document.getElementById("student-info").innerHTML =
                `Имя: ${student.getName()}, Возраст: ${student.getAge()}, Средний балл: ${student.getGrade()}`;
        } else {
            document.getElementById("student-info").innerHTML = "Информация о студенте отсутствует.";
        }
    }

    function calculate(operation) {
        var num1 = parseFloat(document.getElementById("calc-num1").value);
        var num2 = parseFloat(document.getElementById("calc-num2").value);
        var calculator = new Calculator();
        var result;

        try {
            switch(operation) {
                case 'add':
                    result = calculator.add(num1, num2);
```

```
                break;
            case 'subtract':
                result = calculator.subtract(num1, num2);
                break;
            case 'multiply':
                result = calculator.multiply(num1, num2);
                break;
            case 'divide':
                result = calculator.divide(num1, num2);
                break;
        }
        document.getElementById("calculator-result").innerHTML = `Результат: ${result}`;
    } catch (e) {
        document.getElementById("calculator-result").innerHTML = `Ошибка: ${e.message}`;
    }
}

</script>
</body>
</html>
```

figure.js:

```
// Класс Figure
function Figure() {}

Figure.prototype.area = function() {
  throw new Error("This method should be implemented in subclass");
```

```
};

Figure.prototype.perimeter = function() {
  throw new Error("This method should be implemented in subclass");
};

// Класс Rectangle
function Rectangle(width, height) {
  Figure.call(this);
  this.width = width;
  this.height = height;
}

Rectangle.prototype = Object.create(Figure.prototype);
Rectangle.prototype.constructor = Rectangle;

Rectangle.prototype.area = function() {
  return this.width * this.height;
};

Rectangle.prototype.perimeter = function() {
  return 2 * (this.width + this.height);
};

// Класс Circle
function Circle(radius) {
```

```
  Figure.call(this);
  this.radius = radius;
}

Circle.prototype = Object.create(Figure.prototype);
Circle.prototype.constructor = Circle;

Circle.prototype.area = function() {
  return Math.PI * this.radius * this.radius;
};

Circle.prototype.perimeter = function() {
  return 2 * Math.PI * this.radius;
};
```

student.js:

```
// Определение класса Student
class Student {
  #name;
  #age;
  #grade;

  constructor(name, age, grade) {
    this.#name = name;
    this.#age = age;
```

```
  this.#grade = grade;
}

getName() {
  return this.#name;
}

setName(name) {
  this.#name = name;
}

getAge() {
  return this.#age;
}

setAge(age) {
  this.#age = age;
}

getGrade() {
  return this.#grade;
}

setGrade(grade) {
  this.#grade = grade;
}
```

```
}
```

calculator.js:

```
// Определение класса Calculator
class Calculator {
  add(a, b) {
    return a + b;
  }

  subtract(a, b) {
    return a - b;
  }

  multiply(a, b) {
    return a * b;
  }

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

Вывод.
В ходе выполнения данной работы были успешно разработаны и реализованы различные классы в JavaScript, что способствовало углублению понимания концепций объектно-ориентированного программирования. Практика с прототипами и наследованием, особенно в разработке классов "Фигура", "Прямоугольник", "Круг", а также внедрение инкапсуляции в классе "Студент" и создание функционального класса "Калькулятор", демонстрируют приобретенные навыки в работе с ключевыми аспектами объектно-ориентированного программирования в JavaScript.