Лабораторная работа № 5

Тема: Классы в JavaScript.

Цель: Закрепить знания о синтаксисе классов, конструкторе, геттерах/сеттерах и статических свойствах и методах.

🔷 Вариант 1.

1. Создание класса.
Создать класс Rectangle с конструктором (ширина, высота).
Добавить метод area(), который возвращает площадь.
Создать несколько объектов и вывести площадь.

2. Геттер и сеттер.
В классе Rectangle реализовать:
геттер perimeter, который вычисляет периметр,
сеттер size, который принимает массив [width, height] и обновляет размеры.

3. Статические методы.
Добавить статический метод compare(a, b), который сравнивает площади двух прямоугольников.
Проверить работу метода.

🔷 Вариант 2

1. Создание класса.
Создать класс Student с полями name и year (год поступления).
Добавить метод getInfo(), который выводит имя и год.

2. Геттер и сеттер.
Реализовать:
геттер course, который вычисляет текущий курс студента (на основе текущего года),
сеттер course, который изменяет year (например, установка курса 3 → автоматически меняет год поступления).

3. Статические методы.
Добавить статическое свойство university = "MIT".
Добавить статический метод isSameYear(student1, student2), который проверяет, совпадает ли у студентов год поступления.

⚡ **Отчет должен содержать (см. образец):**
- номер и тему лабораторной работы;
- фамилию, номер группы студента и вариант задания;
- скриншоты окна VSC с исходным кодом программ;
- скриншоты с результатами выполнения программ;
- пояснения, если необходимо;
- выводы.

Отчеты в формате **pdf** отправлять на email:
colledge20education23@gmail.com

📌 **Шпаргалка: Классы в JavaScript**

🔷 **1. Определение класса**

```
class Car {
  constructor(brand, model, year) {
    this.brand = brand;   // свойства
    this.model = model;
    this.year = year;
  }

  // метод экземпляра
  getInfo() {
    return `${this.brand} ${this.model} (${this.year})`;
  }
}

let car = new Car("Toyota", "Corolla", 2020);
console.log(car.getInfo()); // Toyota Corolla (2020)
```

constructor() инициализирует объект при new.

Методы пишутся внутри тела класса.

🔷 **2. Геттеры и сеттеры**

```
class Car {
  constructor(brand, model, year) {
    this.brand = brand;
    this.model = model;
    this.year = year;
  }

  // геттер
```

```
  get age() {
    return new Date().getFullYear() - this.year;
  }

  // сеттер
  set age(value) {
    this.year = new Date().getFullYear() - value;
  }
}

let car = new Car("BMW", "X5", 2018);
console.log(car.age);  // 7 (если 2025 год)

car.age = 3;           // переписываем "возраст"
console.log(car.year); // 2022
```

get позволяет обращаться к методу как к свойству.

set помогает контролировать изменение свойств.

🔷 3. Статические свойства и методы

```
class Car {
  constructor(brand, model) {
    this.brand = brand;
    this.model = model;
  }

  getInfo() {
    return `${this.brand} ${this.model}`;
  }

  // статическое свойство
  static type = "vehicle";

  // статический метод
  static compare(c1, c2) {
    return c1.brand.localeCompare(c2.brand);
  }
}

let c1 = new Car("Audi", "A4");
let c2 = new Car("Mercedes", "C-Class");
```

```
console.log(Car.type);              // vehicle
console.log(Car.compare(c1, c2));   // -1 (Audi < Mercedes)
```

Статические члены принадлежат классу, а не объекту.

Удобно для «общей логики» (сравнение, константы и т.д.).

🔷 4. Мини-пример

```
class Book {
  constructor(title, author, year) {
    this.title = title;
    this.author = author;
    this.year = year;
  }

  get summary() {
    return `${this.title} (${this.year}), автор:
${this.author}`;
  }

  static category = "Literature";
}

let b = new Book("1984", "George Orwell", 1949);
console.log(b.summary);          // 1984 (1949), автор:
George Orwell
console.log(Book.category);      // Literature
```