Лабораторная работа № 1

Тема: Основы ООП и инкапсуляция.

Цель: закрепить знания о классах, объектах и принципе инкапсуляции в JavaScript.

Вариант 1

Задание 1. Создание класса.

Создать класс Book с полями: title, author, year.

Написать метод getInfo(), который возвращает строку вида:

"Название: <title>, Автор: <author>, Год: <year>".

Задание 2. Инкапсуляция через приватные свойства.

Добавить в класс Book приватное свойство #pages.

Создать геттер и сеттер для этого свойства:

геттер возвращает количество страниц,

сеттер изменяет его только если значение больше нуля.

Задание 3. Работа с экземплярами.

Создать два экземпляра класса Book.

Установить количество страниц через сеттер.

Вывести информацию о книгах с помощью метода getInfo().

Задание 4. Метод для проверки возраста книги.

Добавить метод isOld(), который возвращает true, если книге больше 20 лет, иначе — false.

Задание 5. Массовая работа с объектами.

Создать массив из 3–5 объектов Book.

Вывести информацию о всех книгах.

Найти и вывести только те книги, у которых количество страниц больше 200.

Вариант 2

Задание 1. Создание класса.

Создать класс Car с полями: brand, model, year.

Написать метод getInfo(), который возвращает строку вида:

"Марка: <brand>, Модель: <model>, Год: <year>".

Задание 2. Инкапсуляция через приватные свойства.

Добавить в класс Car приватное свойство #mileage.

Создать геттер и сеттер:

геттер возвращает пробег,

сеттер увеличивает пробег только если новое значение больше текущего.

Задание 3. Работа с экземплярами.

Создать два экземпляра класса Car.

Установить пробег через сеттер.

Вывести информацию об автомобилях с помощью метода getInfo().

Задание 4. Метод для проверки возраста машины.

Добавить метод isOld(), который возвращает true, если машине больше 10 лет, иначе — false.

Задание 5. Массовая работа с объектами.

Создать массив из 3–5 объектов Car.

Вывести информацию обо всех автомобилях.

Найти и вывести только те машины, у которых пробег больше 100 000 км.

⚡ **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 по классам, объектам и инкапсуляции в JavaScript.

1. Класс и объект
Класс — это шаблон для создания объектов.
Объект (экземпляр) — конкретный представитель класса.

```
// Объявление класса
class User {
  constructor(name, age) {     // конструктор — вызывается при создании объекта
    this.name = name;          // свойство объекта
    this.age = age;
  }

  sayHello() {                 // метод объекта
    console.log(`Привет, меня зовут ${this.name}`);
  }
}

// Создание объектов (экземпляров класса)
const u1 = new User("Иван", 20);
const u2 = new User("Анна", 25);

// вызов методов у объектов
u1.sayHello(); // Привет, меня зовут Иван
u2.sayHello(); // Привет, меня зовут Анна
```

2. Свойства в классе

В конструкторе (уникальные для каждого объекта):

```
class User {
  constructor(name) {
    this.name = name; // задаётся при создании объекта
```

```
  }
}
```

Вне конструктора (значения по умолчанию):

```
class User {
  role = "guest"; // общее значение по умолчанию
}
```

3. Инкапсуляция

Инкапсуляция — принцип, при котором данные объекта защищены от прямого доступа извне.

В JavaScript это реализуется через приватные поля и геттеры/сеттеры.

Приватные свойства.

```
class Account {
  #balance = 0; // приватное свойство (нельзя обратиться напрямую)

  // геттер (чтение)
  get balance() {
    return this.#balance;
  }

  // сеттер (изменение)
  set balance(amount) {
    if (amount >= 0) {
      this.#balance = amount;
    } else {
      console.log("Баланс не может быть отрицательным!");
    }
  }
}
const acc = new Account();
acc.balance = 100;        // используем сеттер
console.log(acc.balance); // 100 (через геттер)
acc.balance = -50;        // Баланс не может быть отрицательным!
```

4. Правила именования геттеров и сеттеров

Названия совпадают с названием свойства, которое они описывают.

Префиксы get/set в названии метода внутри класса не пишутся.

Пример:

```
class User {
  #age = 0;
```

```
  get age() {
    return this.#age;
  }

  set age(value) {
    if (value > 0) this.#age = value;
  }
}

5. Работа с массивом объектов
class Car {
  constructor(brand, year) {
    this.brand = brand;
    this.year = year;
  }
}

const cars = [
  new Car("Toyota", 2010),
  new Car("BMW", 2018),
  new Car("Lada", 2005),
];

// вывод всех машин
cars.forEach(car => console.log(car.brand, car.year));

// фильтр — только новые машины
const newCars = cars.filter(car => car.year > 2015);
console.log(newCars);
```