МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ
Ордена трудового Красного Знамени федеральное государственное бюджетное
образовательное учреждение высшего образования
«Московский технический университет связи и информатики»

Кафедра Математическая кибернетика и информационные технологии

Отчет по лабораторной работе № 2

Выполнил: студент группы БВТ2402

Косякова Олеся Дмитриевна

Москва, 2025

Цель работы: Изучение и практическое применение основных принципов объектно-ориентированного программирования (ООП) на языке Java через создание иерархии классов, демонстрацию инкапсуляции, наследования, полиморфизма и абстракции.

Задание: Создайте иерархию классов в соответствии с вариантом. Ваша иерархия должна содержать:

• абстрактный класс;

• два уровня наследуемых классов (классы должны содержать в себе минимум 3 поля и 2 метода, описывающих поведение объекта);

• демонстрацию реализации всех принципов ООП;

• наличие конструкторов (в том числе по умолчанию);

• наличие геттеров и сеттеров;

• ввод/вывод информации о создаваемых объектах;

• предусмотрите в одном из классов создание счетчика созданных объектов с использованием статической переменной, продемонстрируйте работу.

Базовый класс: Бытовая техника. Дочерние классы: Холодильник, Посудомоечная машина, Пылесос.

Ход работы:

Создаем абстрактный класс, который будет выделять наиболее значимые характеристики объекта. В нем создаем два абстрактных метода, которые будут реализованы в дочерних классах. Создаем конструктор по умолчанию и конструктор с данными, что позволит нам либо сначала задать пустой объект и потом внести данные, либо сразу создать объект с данными. Создаем статическую переменную для подсчета созданных объектов.

```java
abstract class Appliances {
    protected String brand;
    protected String model;
    protected double price;

    // Статическая переменная для подсчета созданных объектов
    private static int appliancesCount = 0;

    // Конструктор по умолчанию
    public Appliances() {
        appliancesCount++;
    }

    // Конструктор с параметрами
    public Appliances(String brand, String model, double price) {
        this.brand = brand;
        this.model = model;
        this.price = price;
        appliancesCount++;
    }

    // Абстрактные методы
    public abstract void turnOn();
    public abstract void turnOff();
```

Создаем геттеры и сеттеры для защиты данных от прямого доступа. Они позволяют получать данные из готовых объектов и изменять их согласно указанным ограничениям.

```java
    //Геттеры и сеттеры
    public String getBrand() {
        return brand;
    }

    public String getModel() {
        return model;
    }

    public double getPrice() {
        return price;
    }

    public void setBrand(String brand) {
        this.brand = brand;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public void setPrice(double price) {
        if (price >= 0) {
            this.price = price;
        }
    }

    public static int getAppliancesCount() {
        return appliancesCount;
    }
```

Создаем дочерний класс Холодильник. Конструктор по умолчанию благодаря вызову super увеличивает счетчик созданных объектов на 1. В конструктор с данными через super передаем свойства родительского класса Appliances. Благодаря ключевому слову this передаем значения объектов в параметры. productsNum задаем явное начальное значение 0. Далее переопределяем методы заданные в родительском классе.

```
class Fridge extends Appliances {
    private double temp;
    private int productsNum;
    private double volume;

    public Fridge() {
        super();
    }

    //Конструктор
    public Fridge(String brand, String model, double price, double temp, double volume) {
        super(brand, model, price);
        this.temp = temp;
        this.productsNum = 0;
        this.volume = volume;
    }

    //Переопределение методов
    @Override
    public void turnOn() {
        System.out.println("Холодильник: " + model + " - включен");
    }
```

Пример реализации статического полиморфизма. Два метода имеет одно и то же имя, но разные списки данных.

```
//Статический полиморфизм
public void freeze(String product) {
    System.out.println("Холодильник: " + model + " заморозил продукт: " + product);
}

 public void freeze(String product, double time) {
    System.out.println("Холодильник: " + model + " заморозил продукт: " + product + " за " + time + " минут");
}
```

По такому же принципу создаем дочерние классы посудомоечная машина и пылесос.

Создаем публичный класс, в котором создаем объекты через метод main.

Пример реализации динамического полиморфизма. Создаем массив appliancesList, в котором заданы объекты дочерних классов. Далее с

помощью цикла вызываем метод каждого объект из массива. Вызвав один и тот же метод у нас получится три разных вывода.

```
Appliances[] appliancesList = {fridge, vacuumCleaner, dishwasher};
for (var i = 0; i < appliancesList.length; i++) {
    appliancesList[i].turnOff();
}
```

Реализуем ввод данных от пользователя

```
// Ввод данных от пользователя
System.out.println(x:"Установите количество тарелок");
int dishNum = console.nextInt();
console.nextLine();
Dishwasher userDish = new Dishwasher();
 userDish.setBrand(brand:"Dyson");
 userDish.setModel(model:"234-90F");
 userDish.setPrice(price:100000);
userDish.setDishesNum(dishNum);
userDish.setProgram(program:"Интенсивная");
userDish.setWaterUse(waterUse:12);
```

Вывод консоли:

```
Холодильник
Компания: LG Модель: GN-B222SLC Цена: 45000.0
Температура: -18.0°C
Объем: 320.0 л
Продуктов: 15
В холодильнике: GN-B222SLC количестов продуктов:16
Холодильник: GN-B222SLC заморозил продукт: арбуз
Холодильник: GN-B222SLC заморозил продукт: арбуз за 5.0 минут

Пылесос
Компания: Samsung Модель: VCC45S0S3R Цена: 25000.0
Тип: Моющий
Мощность: 2000 Вт
Управление: Дистанционное
Включена влажная уборка
Включен производительный режим, мощность увеличина до 2100 Вт

Посудомоечная машина
Компания: Bosch Модель: SMV25AX00R Цена: 38000.0
Количество посуды: 12 комплектов
Программа: Интенсивная
Расход воды: 9.5 л
Выбрана программа: Интенсивная
Посудомоечная машина готова к запуску


Холодильник: GN-B222SLC - выключен
Пылесос: VCC45S0S3R - выключен
Посудомоечная машина: SMV25AX00R - выключена

Установите количество тарелок
35
Объект создан
Выбрана программа: Интенсивная
Посудомоечная машина готова к запуску
Всего создано объектов: 4
```

Вывод: в ходе лабораторной работы мы изучили и практически применили основные принципы объектно-ориентированного программирования (ООП) на языке Java через создание иерархии классов, демонстрацию инкапсуляции, наследования, полиморфизма и абстракции.

Ссылка на репозиторий: https://github.com/swydiz/information_technology