Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №3**

**Дисциплина: Платформо-независимое программирование**

Работу выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.С. Агаджанян

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В. И. Шиян

**Тема**: РАЗРАБОТКА И ИСПОЛЬЗОВАНИЕ ИНТЕРФЕЙСОВ

**Цель**: ознакомиться с понятием интерфейса, правилами разработки, наследования и способами прикладного использования интерфейсов в Java.

**Задание**:

Ориентируясь на диаграмму из примера 3.2, выполните следующие действия:

1. В вашем варианте задачи из ЛР №2 замените базовый класс

интерфейсом. Если позволяет задача, разбейте его на два интерфейса, в качестве примера используйте листинги 12, 13, 14.

2. Разработайте UML-диаграмму и программу для созданной

иерархии.

3. Приведите UML-диаграмму и программу в отчете, добавьте

скриншоты исполнения программы. Классы должны находиться в

разных файлах.

Изображение выглядит как текст, Шрифт, линия, число

Контент, сгенерированный ИИ, может содержать ошибки.

**DeviceBase:**

public interface DeviceBase {

    String getManufacturer();

    void setManufacturer(String manufacturer);

    float getPrice();

    void setPrice(float price);

    String getSerialNumber();

    void setSerialNumber(String serialNumber);

}

**Driver:**

public interface Driver {

    void installDriver();

    void removeDriver();

    void replace();

    void recognize();

}

**EthernetAdapter:**

import java.util.Objects;

public class EthernetAdapter implements DeviceBase, Driver {

    private String manufacturer;

    private float price;

    private String serialNumber;

    private int speed;

    private String mac;

    public EthernetAdapter(String manufacturer, float price, String serialNumber, int speed, String mac) {

        this.manufacturer = manufacturer;

        this.price = price;

        this.serialNumber = serialNumber;

        this.speed = speed;

        this.mac = mac;

    }

    @Override

    public String getManufacturer() {

        return manufacturer;

    }

    @Override

    public void setManufacturer(String manufacturer) {

        this.manufacturer = manufacturer;

    }

    @Override

    public float getPrice() {

        return price;

    }

    @Override

    public void setPrice(float price) {

        this.price = price;

    }

    @Override

    public String getSerialNumber() {

        return serialNumber;

    }

    @Override

    public void setSerialNumber(String serialNumber) {

        this.serialNumber = serialNumber;

    }

    public int getSpeed() {

        return speed;

    }

    public void setSpeed(int speed) {

        this.speed = speed;

    }

    public String getMac() {

        return mac;

    }

    public void setMac(String mac) {

        this.mac = mac;

    }

    @Override

    public void replace() {

        System.out.println("Ethernet Adapter replaced.");

    }

    @Override

    public void recognize() {

        System.out.println("Ethernet Adapter recognized.");

    }

    @Override

    public void installDriver() {

        System.out.println("Ethernet Adapter driver installed.");

    }

    @Override

    public void removeDriver() {

        System.out.println("Ethernet Adapter driver removed.");

    }

    @Override

    public boolean equals(Object o) {

        if (this == o) return true;

        if (o == null || getClass() != o.getClass()) return false;

        EthernetAdapter that = (EthernetAdapter) o;

        return Float.compare(that.getPrice(), getPrice()) == 0 &&

               speed == that.speed &&

               Objects.equals(getManufacturer(), that.getManufacturer()) &&

               Objects.equals(getSerialNumber(), that.getSerialNumber()) &&

               Objects.equals(mac, that.mac);

    }

    @Override

    public int hashCode() {

        return Objects.hash(getManufacturer(), getPrice(), getSerialNumber(), speed, mac);

    }

    @Override

    public String toString() {

        return "EthernetAdapter{" +

               "manufacturer='" + getManufacturer() + '\'' +

               ", price=" + getPrice() +

               ", serialNumber='" + getSerialNumber() + '\'' +

               ", speed=" + speed +

               ", mac='" + mac + '\'' +

               '}';

    }

}

**Monitor:**

import java.util.Objects;

public class Monitor implements DeviceBase, Driver {

    private String manufacturer;

    private float price;

    private String serialNumber;

    private int resolutionX;

    private int resolutionY;

    public Monitor(String manufacturer, float price, String serialNumber, int resolutionX, int resolutionY) {

        this.manufacturer = manufacturer;

        this.price = price;

        this.serialNumber = serialNumber;

        this.resolutionX = resolutionX;

        this.resolutionY = resolutionY;

    }

    @Override

    public String getManufacturer() {

        return manufacturer;

    }

    @Override

    public void setManufacturer(String manufacturer) {

        this.manufacturer = manufacturer;

    }

    @Override

    public float getPrice() {

        return price;

    }

    @Override

    public void setPrice(float price) {

        this.price = price;

    }

    @Override

    public String getSerialNumber() {

        return serialNumber;

    }

    @Override

    public void setSerialNumber(String serialNumber) {

        this.serialNumber = serialNumber;

    }

    public int getResolutionX() {

        return resolutionX;

    }

    public void setResolutionX(int resolutionX) {

        this.resolutionX = resolutionX;

    }

    public int getResolutionY() {

        return resolutionY;

    }

    public void setResolutionY(int resolutionY) {

        this.resolutionY = resolutionY;

    }

    @Override

    public void replace() {

        System.out.println("Monitor replaced.");

    }

    @Override

    public void recognize() {

        System.out.println("Monitor recognized.");

    }

     @Override

    public void installDriver() {

        System.out.println("Monitor driver installed.");

    }

    @Override

    public void removeDriver() {

        System.out.println("Monitor driver removed.");

    }

    @Override

    public boolean equals(Object o) {

        if (this == o) return true;

        if (o == null || getClass() != o.getClass()) return false;

        Monitor monitor = (Monitor) o;

        return Float.compare(monitor.getPrice(), getPrice()) == 0 &&

               resolutionX == monitor.resolutionX &&

               resolutionY == monitor.resolutionY &&

               Objects.equals(getManufacturer(), monitor.getManufacturer()) &&

               Objects.equals(getSerialNumber(), monitor.getSerialNumber());

    }

    @Override

    public int hashCode() {

        return Objects.hash(getManufacturer(), getPrice(), getSerialNumber(), resolutionX, resolutionY);

    }

    @Override

    public String toString() {

        return "Monitor{" +

               "manufacturer='" + getManufacturer() + '\'' +

               ", price=" + getPrice() +

               ", serialNumber='" + getSerialNumber() + '\'' +

               ", resolutionX=" + resolutionX +

               ", resolutionY=" + resolutionY +

               '}';

    }

}

**Main:**

import java.util.ArrayList;

import java.util.List;

public class Main {

    public static void main(String[] args) {

        List<DeviceBase> devices = new ArrayList<>();

        EthernetAdapter adapter1 = new EthernetAdapter("Intel", 50.0f, "67890", 1000, "00:11:22:33:44:55");

        Monitor monitor1 = new Monitor("LG", 200.0f, "ABCDE", 1920, 1080);

        // Добавление уникальных объектов

        if (!devices.contains(adapter1)) {

            devices.add(adapter1);

        }

        if (!devices.contains(monitor1)) {

            devices.add(monitor1);

        }

        // Вывод характеристик каждого объекта

        for (DeviceBase device : devices) {

            System.out.println(device);

            System.out.println("--------------------");

        }

    }

}