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 информационным технологиям и программированию

на тему: **«Объектно-ориентированное программирование»**

Выполнил: студент группы БПИ2403

ФИО: Сон Владимир Сергеевич

Руководитель: Рыбаков Егор Дмитриевич

Москва, 2025

Цель работы: Изучить и применить на практике основные принципы объектно-ориентированного программирования в Java — инкапсуляцию, наследование, полиморфизм и абстракцию.

Задание 1: Создайте иерархию классов в соответствии с вариантом.

```java
import java.util.Scanner;


abstract class ComputerPeripheral {
    protected String brand;
    protected String model;
    protected boolean isWireless;

    public ComputerPeripheral() {
        this.brand = "Неизвестно";
        this.model = "Неизвестно";
        this.isWireless = false;
    }

    public ComputerPeripheral(String brand, String model, boolean isWireless) {
        this.brand = brand;
        this.model = model;
        this.isWireless = isWireless;
    }

    public abstract void connect();
    public abstract void disconnect();

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

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

    public boolean isWireless() { return isWireless; }
    public void setWireless(boolean wireless) { isWireless = wireless; }

    public void displayInfo() {
        System.out.println("Бренд: " + brand + ", Модель: " + model + ", Беспроводной: " + getWirelessStatus());
    }

    private String getWirelessStatus() {
        if (isWireless) {
            return "Да";
        } else {
            return "Нет";
        }
    }
}
```

```java
class Keyboard extends ComputerPeripheral {
    private String layout;
    private boolean hasBacklight;
    private static int keyboardCount = 0;

    public Keyboard() {
        super();
        this.layout = "QWERTY";
        this.hasBacklight = false;
        keyboardCount++;
    }

    public Keyboard(String brand, String model, boolean isWireless,
                    String layout, boolean hasBacklight) {
        super(brand, model, isWireless);
        this.layout = layout;
        this.hasBacklight = hasBacklight;
        keyboardCount++;
    }

    @Override
    public void connect() {
        if (isWireless) {
            System.out.println("Клавиатура подключена по Bluetooth");
        } else {
            System.out.println("Клавиатура подключена по USB");
        }
    }

    @Override
    public void disconnect() {
        System.out.println("Клавиатура отключена");
    }

    public void type() {
        System.out.println("Печатаем на клавиатуре с раскладкой " + layout);
    }

    public void toggleBacklight() {
        hasBacklight = !hasBacklight;
        if (hasBacklight) {
            System.out.println("Подсветка ВКЛЮЧЕНА");
        } else {
            System.out.println("Подсветка ВЫКЛЮЧЕНА");
        }
    }
```

```java
    public static int getKeyboardCount() {
        return keyboardCount;
    }

    public String getLayout() { return layout; }
    public void setLayout(String layout) { this.layout = layout; }

    public boolean hasBacklight() { return hasBacklight; }
    public void setHasBacklight(boolean hasBacklight) { this.hasBacklight = hasBacklight; }

    @Override
    public void displayInfo() {
        super.displayInfo();
        System.out.println("Раскладка: " + layout + ", Подсветка: " + getBacklightStatus());
    }

    private String getBacklightStatus() {
        if (hasBacklight) {
            return "Да";
        } else {
            return "Нет";
        }
    }
}

class Headphones extends ComputerPeripheral {
    private String type;
    private boolean hasNoiseCancellation;

    public Headphones() {
        super();
        this.type = "Накладные";
        this.hasNoiseCancellation = false;
    }

    public Headphones(String brand, String model, boolean isWireless, String type, boolean hasNoiseCancellation) {
        super(brand, model, isWireless);
        this.type = type;
        this.hasNoiseCancellation = hasNoiseCancellation;
    }
```

```java
@Override
public void connect() {
    if (isWireless) {
        System.out.println("Наушники подключены по Bluetooth");
    } else {
        System.out.println("Наушники подключены через разъем");
    }
}

@Override
public void disconnect() {
    System.out.println("Наушники отключены");
}

public void playAudio() {
    System.out.println("Воспроизводим аудио через " + type + " наушники");
}

public void toggleNoiseCancellation() {
    if (hasNoiseCancellation) {
        System.out.println("Шумоподавление переключено");
    } else {
        System.out.println("Шумоподавление не поддерживается");
    }
}

public String getType() { return type; }
public void setType(String type) { this.type = type; }

public boolean hasNoiseCancellation() { return hasNoiseCancellation; }
public void setHasNoiseCancellation(boolean hasNoiseCancellation) {
    this.hasNoiseCancellation = hasNoiseCancellation;
}

@Override
public void displayInfo() {
    super.displayInfo();
    System.out.println("Тип: " + type + ", Шумоподавление: " + getNoiseCancelStatus());
}

private String getNoiseCancelStatus() {
    if (hasNoiseCancellation) {
        return "Да";
    } else {
        return "Нет";
    }
}
```

```java
class GraphicsTablet extends ComputerPeripheral {
    private int pressureLevels;
    private double activeArea;
    private String penType;

    public GraphicsTablet() {
        super();
        this.pressureLevels = 1024;
        this.activeArea = 8.0;
        this.penType = "Без батареи";
    }

    public GraphicsTablet(String brand, String model, boolean isWireless,
                          int pressureLevels, double activeArea, String penType) {
        super(brand, model, isWireless);
        this.pressureLevels = pressureLevels;
        this.activeArea = activeArea;
        this.penType = penType;
    }

    @Override
    public void connect() {
        if (isWireless) {
            System.out.println("Графический планшет подключен по Bluetooth");
        } else {
            System.out.println("Графический планшет подключен по USB");
        }
    }

    @Override
    public void disconnect() {
        System.out.println("Графический планшет отключен");
    }

    public void draw() {
        System.out.println("Рисуем с " + pressureLevels + " уровнями чувствительности");
    }

    public void calibrate() {
        System.out.println("Калибруем активную область " + activeArea + " дюймов");
    }

    public int getPressureLevels() { return pressureLevels; }
    public void setPressureLevels(int pressureLevels) { this.pressureLevels = pressureLevels; }

    public double getActiveArea() { return activeArea; }
    public void setActiveArea(double activeArea) { this.activeArea = activeArea; }
```

```java
    public String getPenType() { return penType; }
    public void setPenType(String penType) { this.penType = penType; }

    @Override
    public void displayInfo() {
        super.displayInfo();
        System.out.println("Уровни давления: " + pressureLevels + ", Активная область: " + activeArea + " дюймов" + ", Тип пера: " + penType);
    }
}


public class ComputerPeripheralDemo {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        ComputerPeripheral[] peripherals = new ComputerPeripheral[3];


        System.out.println("Введите данные клавиатуры:");
        System.out.print("Бренд: ");
        String kbBrand = scanner.nextLine();
        System.out.print("Модель: ");
        String kbModel = scanner.nextLine();
        System.out.print("Беспроводной: ");
        String kbWirelessInput = scanner.nextLine();
        boolean kbWireless;
        if (kbWirelessInput.equalsIgnoreCase("да")) {
            kbWireless = true;
        } else {
            kbWireless = false;
        }
        System.out.print("Раскладка: ");
        String kbLayout = scanner.nextLine();
        System.out.print("Подсветка (да/нет): ");
        String kbBacklightInput = scanner.nextLine();
        boolean kbBacklight;
        if (kbBacklightInput.equalsIgnoreCase("да")) {
            kbBacklight = true;
        } else {
            kbBacklight = false;
        }
        peripherals[0] = new Keyboard(kbBrand, kbModel, kbWireless, kbLayout, kbBacklight);
```

```
        System.out.println("\nВведите данные наушников:");
        System.out.print("Бренд: ");
        String hpBrand = scanner.nextLine();
        System.out.print("Модель: ");
        String hpModel = scanner.nextLine();
        System.out.print("Беспроводной: ");
        String hpWirelessInput = scanner.nextLine();
        boolean hpWireless;
        if (hpWirelessInput.equalsIgnoreCase("да")) {
            hpWireless = true;
        } else {
            hpWireless = false;
        }
        System.out.print("Тип: ");
        String hpType = scanner.nextLine();
        System.out.print("Шумоподавление: ");
        String hpNoiseCancelInput = scanner.nextLine();
        boolean hpNoiseCancel;
        if (hpNoiseCancelInput.equalsIgnoreCase("да")) {
            hpNoiseCancel = true;
        } else {
            hpNoiseCancel = false;
        }
        peripherals[1] = new Headphones(hpBrand, hpModel, hpWireless, hpType, hpNoiseCancel);

        System.out.println("\nВведите данные графического планшета:");
        System.out.print("Бренд: ");
        String gtBrand = scanner.nextLine();
        System.out.print("Модель: ");
        String gtModel = scanner.nextLine();
        System.out.print("Беспроводной: ");
        String gtWirelessInput = scanner.nextLine();
        boolean gtWireless;
        if (gtWirelessInput.equalsIgnoreCase("да")) {
            gtWireless = true;
        } else {
            gtWireless = false;
        }
        System.out.print("Уровни давления: ");
        int gtPressure = scanner.nextInt();
        System.out.print("Активная область: ");
        double gtArea = scanner.nextDouble();
        scanner.nextLine();
        System.out.print("Тип пера: ");
        String gtPen = scanner.nextLine();
        peripherals[2] = new GraphicsTablet(gtBrand, gtModel, gtWireless, gtPressure, gtArea, gtPen);
```

```
        System.out.println("Бренд клавиатуры: " + ((Keyboard)peripherals[0]).getBrand());
        System.out.println("Модель наушников: " + peripherals[1].getModel());

        for (ComputerPeripheral peripheral : peripherals) {
            peripheral.displayInfo();
            peripheral.connect();
            peripheral.disconnect();
            System.out.println();
        }

        Keyboard keyboard = (Keyboard) peripherals[0];
        keyboard.type();
        keyboard.toggleBacklight();

        Headphones headphones = (Headphones) peripherals[1];
        headphones.playAudio();
        headphones.toggleNoiseCancellation();

        GraphicsTablet tablet = (GraphicsTablet) peripherals[2];
        tablet.draw();
        tablet.calibrate();

        Keyboard extraKeyboard = new Keyboard();
        System.out.println("Всего создано клавиатур: " + Keyboard.getKeyboardCount());

        scanner.close();
    }
}
```

Вывод: В ходе работы были успешно освоены и применены на практике основные принципы объектно-ориентированного программирования в Java — инкапсуляцию, наследование, полиморфизм и абстракция.