МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”
КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

ОТЧЁТ

по лабораторной работе №4

Выполнила:
студентка группы ПО-9
Кот А. А.

Проверил:
Крощенко А. А.

Брест 2024

Цель работы: приобрести практические навыки в области объектно-ориентированного проектирования.

Вариант 8.
Ход работы

Задание 1.

Реализовать указанный класс, включив в него вспомогательный внутренний класс или классы. Реализовать 2-3 метода (на выбор). Продемонстрировать использование реализованных классов.
Создать класс CD (mp3-диск) с внутренним классом, с помощью объектов которого можно хранить информацию о каталогах, подкаталогах и записях.

Работа программы:

```
Directory Music not found!
CD Contents:
Directory: The Pretty Reckless
- justtonight.mp3
- you.mp3
Directory: Muse
- hysteria.mp3

Process finished with exit code 0
```

Код программы:

CD.java

```
import java.util.ArrayList;
import java.util.List;

public class CD {
    private List<Directory> directories;

    public CD() {
        this.directories = new ArrayList<>();
    }

    private class Directory {
        private String name;
        private List<String> files;

        public Directory(String name) {
            this.name = name;
            this.files = new ArrayList<>();
        }

        public void addFile(String file) {
            this.files.add(file);
        }

        public void printDirectory() {
            System.out.println("Directory: " + name);
            for (String file : files) {
```

```
                System.out.println("- " + file);
            }
        }
    }

    public void addDirectory(String name) {
        directories.add(new Directory(name));
    }

    public void addFileToDirectory(String directoryName, String fileName) {
        for (Directory directory : directories) {
            if (directory.name.equals(directoryName)) {
                directory.addFile(fileName);
                return;
            }
        }
        System.out.println("Directory " + directoryName + " not found!");
    }

    public void printCDContents() {
        System.out.println("CD Contents:");
        for (Directory directory : directories) {
            directory.printDirectory();
        }
    }

    public static void main(String[] args) {
        CD cd = new CD();

        cd.addDirectory("The Pretty Reckless");
        cd.addFileToDirectory("The Pretty Reckless", "justtonight.mp3");
        cd.addFileToDirectory("The Pretty Reckless", "you.mp3");

        cd.addDirectory("Muse");
        cd.addFileToDirectory("Muse", "hysteria.mp3");
        cd.addFileToDirectory("Music", "song.mp3");

        cd.printCDContents();
    }
}
```

Задание 2.

Реализовать агрегирование. При создании класса агрегируемый класс объявляется как атрибут (локальная переменная, параметр метода). Включить в каждый класс 2-3 метода на выбор. Продемонстрировать использование разработанных классов.

Создать класс Текст, используя класс Абзац.

Работа программы:

```
Количество абзацев: 2
Количество слов: 6
Содержимое текста:
Это первый абзац.
Это второй абзац.

Process finished with exit code 0
```

Код программы:

Task2Main.java

```
public class Task2Main {
    public static void main(String[] args) {
        Paragraph paragraph1 = new Paragraph("Это первый абзац.");
        Paragraph paragraph2 = new Paragraph("Это второй абзац.");

        Text text = new Text();
        text.addParagraph(paragraph1);
        text.addParagraph(paragraph2);

        System.out.println("Количество абзацев: " + text.countParagraphs());
        System.out.println("Количество слов: " + text.countWords());

        System.out.println("Содержимое текста:");
        text.display();
    }
}
```

Paragraph.java

```
public class Paragraph {
    private String content;

    public Paragraph(String content) {
        this.content = content;
    }

    public String getContent() {
        return content;
    }

    public void setContent(String content) {
        this.content = content;
    }

    public int countWords() {
        String[] words = content.split("\\s+");
        return words.length;
    }
}
```

Text.java

```
import java.util.ArrayList;
import java.util.List;

public class Text {
    private List<Paragraph> paragraphs;

    public Text() {
        this.paragraphs = new ArrayList<>();
    }

    public void addParagraph(Paragraph paragraph) {
        paragraphs.add(paragraph);
    }

    public void removeParagraph(Paragraph paragraph) {
        paragraphs.remove(paragraph);
    }
```

```
    public int countParagraphs() {
        return paragraphs.size();
    }

    public int countWords() {
        int totalWords = 0;
        for (Paragraph paragraph : paragraphs) {
            totalWords += paragraph.countWords();
        }
        return totalWords;
    }

    public void display() {
        for (Paragraph paragraph : paragraphs) {
            System.out.println(paragraph.getContent());
        }
    }
}
```

Задание 3.

Построить модель программной системы с применением отношений (обобщения, агрегации, ассоциации, реализации) между классами. Задать атрибуты и методы классов. Реализовать (если необходимо) дополнительные классы. Продемонстрировать работу разработанной системы.

Система Интернет-магазин. Администратор добавляет информацию о Товаре. Клиент делает и оплачивает Заказ на Товары. Администратор регистрирует Продажу и может занести неплательщиков в «черный список».

Работа программы:

```
Client: Maria, Amount Paid: $2000.0
Client Vera doesn't have enough money and has been added to the blacklist.
Is Vera blacklisted? true
Laptop Description: High-performance laptop with SSD storage.
Smartphone Description: Latest smartphone with dual-camera setup.

Process finished with exit code 0
```

Код программы:

Task3.java

```
import java.util.ArrayList;
import java.util.List;

class Product {
    private String name;
    private double price;
    private String description;

    public Product(String name, double price) {
        this.name = name;
        this.price = price;
    }

    public String getName() {
        return name;
```

```
    }

    public double getPrice() {
        return price;
    }

    public String getDescription() {
        return description;
    }

    public void setDescription(String description) {
        this.description = description;
    }
}

class Cart {
    private List<Product> products;

    public Cart() {
        products = new ArrayList<>();
    }

    public void addProduct(Product product) {
        products.add(product);
    }

    public double calculateTotalPrice() {
        double total = 0;
        for (Product product : products) {
            total += product.getPrice();
        }
        return total;
    }
}

class Administrator {
    static private List<String> paymentRegistry = new ArrayList<>();
    static private List<String> blacklist = new ArrayList<>();

    public static void registerPayment(String clientName, double amount) {
        paymentRegistry.add("Client: " + clientName + ", Amount Paid: $" +
amount);
        System.out.println("Client: " + clientName + ", Amount Paid: $" +
amount);
    }

    public static void addToBlacklist(String clientName) {
        blacklist.add(clientName);
    }

    public static boolean isBlacklisted(String clientName) {
        return blacklist.contains(clientName);
    }

    public static void addProductDescription(Product product, String
description) {
        product.setDescription(description);
    }

}

class Client {
    private String name;
    private double money;
    private Cart cart;
```

```
    public Client(String name, double money) {
        this.name = name;
        this.money = money;
        this.cart = new Cart();
    }

    public void addProductToCart(Product product) {
        cart.addProduct(product);
    }

    public void pay() {
        double totalPrice = cart.calculateTotalPrice();
        if (totalPrice <= money) {
            money -= totalPrice;
            Administrator.registerPayment(name, totalPrice);
        } else {
            Administrator.addToBlacklist(name);
            System.out.println("Client " + name + " doesn't have enough money
" +
                    "and has been added to the blacklist.");
        }
    }
}

public class Task3{
    public static void main(String[] args) {
        Client client1 = new Client("Maria", 2000);
        Client client2 = new Client("Vera", 50);

        Product product1 = new Product("Laptop", 1200);
        Product product2 = new Product("Smartphone", 800);

        Administrator.addProductDescription(product1,
                "High-performance laptop with SSD storage.");
        Administrator.addProductDescription(product2,
                "Latest smartphone with dual-camera setup.");

        client1.addProductToCart(product1);
        client1.addProductToCart(product2);

        client2.addProductToCart(product1);

        client1.pay(); // Maria has enough money to pay for both products
        client2.pay(); // Vera doesn't have enough money and will be
blacklisted

        // Checking if Vera is blacklisted
        System.out.println("Is Vera blacklisted? " +
                Administrator.isBlacklisted("Vera"));

        // Retrieving product descriptions
        System.out.println(product1.getName() + " Description: " +
product1.getDescription());
        System.out.println(product2.getName() + " Description: " +
product2.getDescription());
    }
}
```

Вывод: в ходе лабораторной работы мы приобрели практические навыки в области объектно-ориентированного проектирования.