Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

ОТЧЕТ

ЛАБОРАТОРНАЯ РАБОТА №4

ССП

Выполнил
студент группы
АС-50:
Протасевич А.В.
Проверил:
Крощенко А.А.

Брест 2020

Цель работы:

приобрести практические навыки в области объектно-ориентированного проектирования.

Вариант 8

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.

Код программы:

Main.java

```
public class main {

    public static void main(String[] args) {

        CD cd = new CD();


        cd.addCatalog("first").setCatalog("first_1").setCatalog("first_2");


cd.addCatalog("second").setCatalog("second_1").setCatalog("second_2").setCata
log("second_3").setCatalog("second_4");



cd.addCatalog("third").setCatalog("third_1").setCatalog("third_2").setCatalog
("third_3");

        cd.print();

    }

}
```

CD.java

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

public class CD {
    private List<Catalog> catalogList;
    public CD() {
        catalogList = new ArrayList<>();
    }
    public Catalog addCatalog(String name) {
        Catalog catalog = new Catalog(name);
        catalog.setSpace("\t");
        catalogList.add(catalog);
        return catalog;
    }
    public void print() {
        catalogList.forEach(System.out::println);
```

```
    }
    class Catalog {
        private String name;
        private Catalog catalog;
        private String space;
        public Catalog(String name) {
            this.name = name;
        }
        public void setSpace(String space) {
            this.space = space;
        }
        public Catalog setCatalog(String name) {
            Catalog catalog = new Catalog(name);
            catalog.setSpace(this.space+"\t");
            this.catalog = catalog;
            return catalog;
        }
        @Override
        public String toString() {
            return name + "\n" + space + (Objects.nonNull(catalog) ? catalog
: "");
        }
    }
}
```

Вывод программы:

```
first
    first_1
        first_2

second
    second_1
        second_2
            second_3
                second_4

third
    third_1
        third_2
            third_3
```

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.

Main.java

```
public class main {

        public static void main(String[] args) {
            Paragraph paragraph1 = new Paragraph();
            paragraph1.addString("Первая строка в первом абзаце.");
            paragraph1.addString("Вторая строка в первом абзаце.");
```

```
            paragraph1.addString("Третья строка в первом абзаце.");
            Paragraph paragraph2 = new Paragraph();
            paragraph2.addString("Первая строка во втором абзаце.");
            paragraph2.addString("Вторая строка во втором абзаце.");
            paragraph2.addString("Третья строка во втором абзаце.");
            Paragraph paragraph3 = new Paragraph();
            paragraph3.addString("Первая строка в третьем абзаце.");
            paragraph3.addString("Вторая строка в третьем абзаце.");
            paragraph3.addString("Третья строка в третьем абзаце.");
            paragraph3.deleteString(2);
            text text = new text();
            text.addParagraph(paragraph1);
            text.addParagraph(paragraph2);
            text.addParagraph(paragraph3);
            text.printText();
            text.deleteParagraph(1);
            System.out.println("После удаления второго абзаца:");
            text.printText();
        }
}
```

Paragraph.java

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

public class Paragraph {
    private List<String> listStrings;
    public Paragraph() {
        this.listStrings = new ArrayList<>();
    }
    public List<String> getListStrings() {
        return listStrings;
    }
    public void addString(String string) {
        listStrings.add(string);
    }
    public void deleteString(int stringNumber) {
        listStrings.remove(stringNumber);
    }
    public void printParagraph() {
        listStrings.forEach(string -> System.out.print(string.concat(" ")));
    }
}
```

text.java

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

public class text {
        private List<Paragraph> listParagraph;
        public text() {
            listParagraph = new ArrayList<>();
        }
        public List<Paragraph> getListParagraph() {
            return listParagraph;
        }
        public void addParagraph(Paragraph paragraph) {
            listParagraph.add(paragraph);
        }
```

```
        public void deleteParagraph(int paragraphId) {
            listParagraph.remove(paragraphId);
        }
        public void printText() {
            listParagraph.forEach(paragraph -> {paragraph.printParagraph();
                System.out.println("\n");});
        }
    }
```

Вывод программы:

```
Первая строка в первом абзаце. Вторая строка в первом абзаце. Третья строка в первом абзаце.

Первая строка во втором абзаце. Вторая строка во втором абзаце. Третья строка во втором абзаце.

Первая строка в третьем абзаце. Вторая строка в третьем абзаце.

После удаления второго абзаца:
Первая строка в первом абзаце. Вторая строка в первом абзаце. Третья строка в первом абзаце.

Первая строка в третьем абзаце. Вторая строка в третьем абзаце.
```

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».

Main.java

```
public class main {
    public static void main(String[] args) {
        OnlineStore onlineStore = new OnlineStore();
        Administrator administrator = new Administrator(onlineStore);
        administrator.addProduct("Book");
        administrator.addProduct("Laptop");
        administrator.addProduct("Lamp");
        administrator.addProduct("Phone");
        administrator.addProduct("Pencil");
        administrator.addProduct("Table");
        Client client1 = new Client(1, onlineStore);
        System.out.println("All products:");
        client1.printProducts();
        client1.addOrder(6).pay();
        client1.addOrder(4);
        client1.addOrder(3).pay();
        System.out.println("\nFirst User orders:");
        client1.printOrders();
        Client client2 = new Client(2, onlineStore);
        client2.addOrder(5).pay();
        client2.addOrder(1).pay();
        System.out.println("\nSecond User orders:");
        client2.printOrders();
```

```
        administrator.addToBlackList();
        System.out.println("\nAdministrator add users to BlackList \n if they
don't pay");
        System.out.println("\nUser is trying to add order:");
        System.out.println("\nFirst User:");
        client1.addOrder(5);
        System.out.println("\nSecond User:");
        client2.addOrder(6).pay();
    }
}
```

Administrator.java

```
public class Administrator {
    private OnlineStore onlineStore;
    public Administrator(OnlineStore onlineStore) {
        this.onlineStore = onlineStore;
    }
    public void addProduct(String productName) {
        onlineStore.addProduct(productName);
    }
    public void addToBlackList() {
        onlineStore.addToBlackList();
    }
}
```

Client.java

```
public class Client {
    private int clientId;
    private OnlineStore onlineStore;
    public Client(int clientId, OnlineStore onlineStore) {
        this.clientId = clientId;
        this.onlineStore = onlineStore;
    }
    public void printProducts() {
        onlineStore.printProducts();
    }
    public void printOrders() {
        onlineStore.printUserOrders(clientId);
    }
    public Order addOrder(int productId) {
        return onlineStore.addOrder(clientId, productId);
    }
}
```

OnlineStore.java

```
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;
public class OnlineStore {
    private List<Product> productList;
    private List<Order> orderList;
    private List<Integer> blackList;
    public OnlineStore() {
        this.productList = new ArrayList<>();
        this.orderList = new ArrayList<>();
        this.blackList = new ArrayList<>();
    }
    private boolean isInBlackList(int userId) {
```

```
        return blackList.contains(userId);
    }
    public void printProducts() {
        productList.forEach(System.out::println);
    }
    public void printUserOrders(int userId) {
        orderList.stream().filter(order -> order.getUserId() ==
                userId).forEach(System.out::println);
    }
    public void addToBlackList() {
        blackList = orderList.stream().filter(order ->

!order.isPaid()).map(Order::getUserId).collect(Collectors.toList());
    }
    public void addProduct(String productName) {
        productList.add(new Product(productList.size() + 1, productName));
    }
    public Order addOrder(int userId, int productId) {
        if (!isInBlackList(userId)) {
            Order order = new Order(userId, productList.get(--productId));
            orderList.add(order);
            return order;
        } else {
            System.out.println("Sorry, but you are in the blacklist");
            return null;
        }
    }
}
```

Order.java

```
public class Order {
    private int userId;
    private Product product;
    private boolean isPaid;

    public Order(int userId, Product product) {
        this.userId = userId;
        this.product = product;
        this.isPaid = false;
    }

    public void pay() {
        this.isPaid = true;
    }

    public int getUserId() {
        return userId;
    }

    public boolean isPaid() {
        return isPaid;
    }

    @Override
    public String toString() {
        return product +
                ", isPaid=" + isPaid;
    }
}
```

Product.java

```
public class Product {
    private int productId;
    private String productName;
    public Product() {
    }
    public Product(int productId, String productName) {
        this.productId = productId;
        this.productName = productName;
    }
    public int getProductId() {
        return productId;
    }
    public void setProductId(int productId) {
        this.productId = productId;
    }
    public String getProductName() {
        return productName;
    }
    public void setProductName(String productName) {
        this.productName = productName;
    }
    @Override
    public String toString() {
        return "Product №" + productId +
                ": '" + productName + '\'';
    }
}
```

Вывод программы:

```
All products:
Product №1: 'Book'
Product №2: 'Laptop'
Product №3: 'Lamp'
Product №4: 'Phone'
Product №5: 'Pencil'
Product №6: 'Table'

First User orders:
Product №6: 'Table', isPaid=true
Product №4: 'Phone', isPaid=false
Product №3: 'Lamp', isPaid=true

Second User orders:
Product №5: 'Pencil', isPaid=true
Product №1: 'Book', isPaid=true

Administrator add users to BlackList
 if they don't pay

User is trying to add order:
```

```
First User:
Sorry, but you are in the blacklist

Second User:
```

Вывод: приобрел практические навыки в области объектно-ориентированного проектирования.