**Министерство образования Республики Беларусь**

**Учреждение образования**

**«Брестский государственный технический университет»**

**Кафедра ИИТ**

**Лабораторная работа №4**

По дисциплине: «ССП»

Вариант 11

**Выполнил:** Студент 3 курса

Группы ПО-8 Замалетдинов Д.А.

**Проверил:**

Крощенко А.А

**Брест 2024**

**Лабораторная работа №4**

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

**Задание 1:** Реализовать указанный класс, включив в него вспомогательный внутренний класс или классы. Реализовать 2-3 метода (на выбор). Продемонстрировать использование реализованных классов.

2) Создать класс Payment (покупка) с внутренним классом, с помощью объектов которого можно сформировать покупку из нескольких товаров.

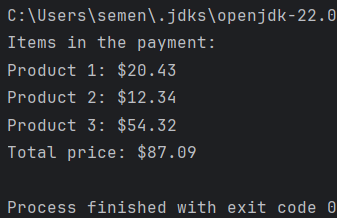
# Код программы: Payment.java

import java.util.ArrayList; import java.util.List;  
  
public class Payment {  
 private List<Item> items;  
  
 public Payment() {  
 this.items = new ArrayList<>();  
 }  
 public void addItem(String name, double price) {  
 Item item = new Item(name, price);  
 items.add(item);  
 }  
 public double getTotalPrice() {  
 double totalPrice = 0;  
 for (Item item : items) {  
 totalPrice += item.getPrice();  
 }  
 return totalPrice;  
 }  
 public void displayItems() {  
 System.*out*.println("Items in the payment:");  
 for (Item item : items) {  
 System.*out*.println(item.getName() + ": $" + item.getPrice());  
 }  
 }  
  
 private class Item {  
 private String name;  
 private double price;  
  
 public Item(String name, double price) {  
 this.name = name;  
 this.price = price;  
 }  
 public String getName() {  
 return name;  
 }  
 public double getPrice() {  
 return price;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
 public void setPrice(double price) {  
 this.price = price;  
 }  
 }  
  
}

# Main.java

public class Main {  
 public static void main(String[] args) {  
 Payment payment = new Payment();  
 payment.addItem("Product 1", 20.43);  
 payment.addItem("Product 2", 12.34);  
 payment.addItem("Product 3", 54.32);  
  
 payment.displayItems();  
 System.*out*.println("Total price: $" + payment.getTotalPrice());  
 } }

**Результат работы программы:**

****

**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.

8) Создать класс Текст, используя класс Абзац.

# Код программы: 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;  
 }  
}

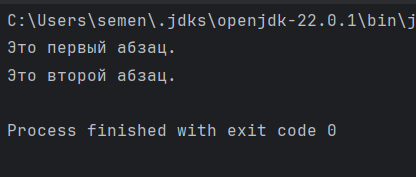
# Text.java

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) {  
 this.paragraphs.add(paragraph);  
 }  
  
 public void printText() {  
 for (Paragraph paragraph : paragraphs) {  
 System.*out*.println(paragraph.getContent());  
 }  
 }  
}

# Main.java

public class Main {  
 public static void main(String[] args) {  
 Paragraph p1 = new Paragraph("Это первый абзац.");  
 Paragraph p2 = new Paragraph("Это второй абзац.");  
  
 Text text = new Text();  
 text.addParagraph(p1);  
 text.addParagraph(p2);  
  
 text.printText();  
 }  
}

**Результаты работы программы:**

****

**Задание 3:** Разработать автоматизированную систему на основе некоторой структуры данных, манипулирующей объектами пользовательского класса. Реализовать требуемые функции обработки данных.

9) Система Железнодорожная касса. Пассажир делает Заявку на станцию назначения, вре-

мя и дату поездки. Система регистрирует Заявку и осуществляет поиск подходящего Поезда.

Пассажир делает выбор Поезда и получает Счет на оплату. Администратор вводит номера

Поездов, промежуточные и конечные станции, цены.

**Код программы:**

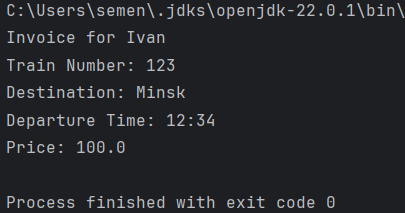
# System.java

import java.util.\*;  
  
class Train {  
 String number;  
 String destination;  
 String departureTime;  
 double price;  
  
 public Train(String number, String destination, String departureTime, double price) {  
 this.number = number;  
 this.destination = destination;  
 this.departureTime = departureTime;  
 this.price = price;  
 }  
}  
  
class Passenger {  
 String name;  
 String destination;  
 String departureTime;  
  
 public Passenger(String name, String destination, String departureTime) {  
 this.name = name;  
 this.destination = destination;  
 this.departureTime = departureTime;  
 }  
}  
  
class RailwayTicketSystem {  
 List<Train> trains = new ArrayList<>();  
 List<Passenger> passengers = new ArrayList<>();  
  
 public void addTrain(String number, String destination, String departureTime, double price) {  
 trains.add(new Train(number, destination, departureTime, price));  
 }  
  
 public void addPassenger(String name, String destination, String departureTime) {  
 passengers.add(new Passenger(name, destination, departureTime));  
 }  
  
 public Train searchTrain(String destination, String departureTime) {  
 for (Train train : trains) {  
 if (train.destination.equals(destination) && train.departureTime.equals(departureTime)) {  
 return train;  
 }  
 }  
 return null;  
 }  
  
 public void printInvoice(Passenger passenger, Train train) {  
 System.*out*.println("Invoice for " + passenger.name);  
 System.*out*.println("Train Number: " + train.number);  
 System.*out*.println("Destination: " + train.destination);  
 System.*out*.println("Departure Time: " + train.departureTime);  
 System.*out*.println("Price: " + train.price);  
 }  
}

# Main.java

class Main {  
 public static void main(String[] args) {  
 RailwayTicketSystem system = new RailwayTicketSystem();  
  
 // Администратор вводит данные о поездах  
 system.addTrain("123", "Minsk", "12:34", 100.0);  
 system.addTrain("456", "Vitebsk", "21:43", 111.0);  
  
 // Пассажир делает заявку  
 system.addPassenger("Ivan", "Minsk", "12:34");  
  
 // Система осуществляет поиск подходящего поезда  
 Train train = system.searchTrain("Minsk", "12:34");  
  
 if (train != null) {  
 // Пассажир выбирает поезд и получает счет на оплату  
 Passenger passenger = system.passengers.get(0);  
 system.printInvoice(passenger, train);  
 } else {  
 System.*out*.println("No suitable train found");  
 }  
 }  
}

**Результаты работы программы:**

****

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