­­­­­­­­Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра автоматизованих систем управління



**Звіт**

до лабораторної роботи № 6

з дисципліни

*Прикладне програмування*

на тему:

**“ Комплексна лабораторна робота ”**

Виконав: студент ОІ-24

**Живоронко Роман**

Прийняв: асистент каф. АСУ

Чорненький В.Я.

**Львів – 2024**

*Лабораторна робота №* *6*

**Завдання лабораторної роботи**

Створити консольну програму на мові програмування Java згідно до індивідуального завдання (ваш номер варіанту, згенерований випадковим чином, можна знайти у файлі «Варіанти завдань.pdf»). Саме завдання описане досить поверхово, якщо ви претендуєте на високу оцінку – його потрібно розширити. Наприклад, у завданні може бути сказано про «створити кілька квіток, з наявних квіток сформувати букет». Цілком логічно буде додати можливість видалити квітку, поміняти параметри існуючої квітки тощо. Оцінка за лабораторну роботу залежатиме і від глибини пропрацювання предметної області.

Окрім цього, програма повинна відповідати таким додатковим вимогам:

• Використовувати можливості ООП: класи, успадкування, поліморфізм, інкапсуляцію.

• Кожний клас повинен мати назву, яка повністю описує його суть, і інформативний склад.

• Успадкування потрібно використовувати тільки тоді, коли воно має сенс.

• При записі програми потрібно використовувати домовленості щодо оформлення коду java code convention.

• Класи потрібно грамотно розкласти по пакетах.

• Для збереження параметрів ініціалізації потрібно використовувати файли.

• Повинне бути консольне меню для вибору різних дій, наприклад: «завантажити з файлу», «довідка», «вихід» тощо.

Етапи виконання завдання (кожен етап вважається окремою лабораторною роботою):

1. Уважно пропрацюйте предметну область згідно вашого варіанту завдання. Опишіть технологічне завдання (який функціонал матиме ваша програма). Створіть Use Case та Class діаграми для вашого завдання.

2. Реалізуйте консольне меню до вашого завдання з використанням паттерну «Команда». На цьому етапі не потрібно реалізовувати функціонал конкретних пунктів меню – тільки створити «кістяк» вашої програми.

3. Реалізуйте функціонал програми. Обов’язково зверніть увагу на додаткові вимоги до програми (описані вище).

4. Покрийте вашу програму юніт тестами (coverage >=90%).

5. Додайте логгер з логуванням основних дій та виняткових ситуацій. При цьому далі логу мають зберігатися у файлі, а критичні помилки – додатково надсилатися на e-mail

**Варіант 8**

Транспорт. Визначити ієрархію рухомого складу залізничного транспорту. Створити пасажирський потяг. Підрахувати загальну кількість пасажирів та багажу. Здійснити сортування вагонів поїзда на основі рівня комфортності. Знайти вагони в поїзді, що відповідають заданому діапазону параметрів кількості пасажирів.

**Код програми**

package Train;  
import Coaches.\*;  
  
  
import java.util.ArrayList;  
import java.util.\*;  
  
public class Train {  
 private String name;  
 private ArrayList <Coach> coaches;  
  
 public Train(String name){  
 setName(name);  
 this.coaches = new ArrayList<>();  
 }  
  
  
 public void setName(String name) {this.name = name;}  
  
 public void addCoach(Coach coach){  
 coaches.add(coach);  
 }  
  
 public int numofpass() {  
 int numOfPass = 0;  
 for(int i = 0;i<11;i++){  
 numOfPass += coaches.get(i).getNumberOfSeats();  
 }  
 return numOfPass;  
 }  
 public int numofbagage(){  
 int numOfBags = 0;  
 for(int i = 0;i<11;i++){  
 numOfBags += (coaches.get(i).getNumOfBagsPerPerson() \* coaches.get(i).getNumberOfSeats());  
 }  
 return numOfBags;  
 }  
 public void sortCoaches() {  
 for (int i = 0; i < coaches.size() - 1; i++) {  
 for (int j = 0; j < coaches.size() - 1 - i; j++) {  
 if (coaches.get(j).getComfLevel() > coaches.get(j + 1).getComfLevel()) {  
 Collections.*swap*(coaches, j, j + 1);  
 }  
 }  
 }  
 }  
  
 public void findCoach(int numOfPl){  
 for(int i = 0; i < coaches.size()-1;i++){  
 if (numOfPl >= coaches.get(i).getNumberOfSeats()){  
 System.*out*.println(coaches.get(i));  
 }  
 }  
 }  
  
 @Override  
 public String toString() {  
 return "\n"+name +  
 ", Вагони:" + coaches;  
 }  
}

package Coaches;  
  
public abstract class Coach {  
 private int numberOfSeats;  
 private String name;  
 private double price;  
 private int numOfBagsPerPerson;  
  
 public Coach(String name,int numberOfSeats,int numOfBagsPerPerson,double price){  
 setName(name);  
 setNumberOfSeats(numberOfSфeats);  
 setNumOfBagsPerPerson(numOfBagsPerPerson);  
 setPrice(price);  
 }  
  
 public abstract int getComfLevel();  
 public int getNumberOfSeats() {return numberOfSeats;}  
 public int getNumOfBagsPerPerson() {return numOfBagsPerPerson;}  
  
 public void setNumberOfSeats(int numberOfSeats) {this.numberOfSeats = numberOfSeats;}  
 public void setName(String name) {this.name = name;}  
 public void setPrice(double price) {this.price = price;}  
 public void setNumOfBagsPerPerson(int numOfBagsPerPerson) {this.numOfBagsPerPerson = numOfBagsPerPerson;}  
  
 @Override  
 public String toString() {  
 return "Назва вагона - " + name +  
 ", К-сть місць - " + numberOfSeats +  
 ", К-сть сумок на особу - " + numOfBagsPerPerson +  
 ", Ціна - " + price + "грн";  
 }  
}

package Coaches;  
  
public class EcoCoach extends Coach {  
 private int comfLevel;  
  
 public EcoCoach(){  
 super("Економ клас",60,2,500);  
 setComfLevel(1);  
 }  
  
 @Override  
 public int getComfLevel() {return comfLevel;}  
  
 public void setComfLevel(int comfLevel) {this.comfLevel = comfLevel;}  
  
 @Override  
 public String toString() {  
 return super.toString() +  
 " Рівень комфортності " + comfLevel + "\n";  
 }  
}

package Coaches;  
  
public class MidleCoach extends Coach {  
 private int comfLevel;  
  
 public MidleCoach(){  
 super("Середній клас",45,4,750);  
 setComfLevel(2);  
 }  
  
 @Override  
 public int getComfLevel() {return comfLevel;}  
  
 public void setComfLevel(int comfLevel) {this.comfLevel = comfLevel;}  
  
 @Override  
 public String toString() {  
 return super.toString() +  
 " Рівень комфортності " + comfLevel + "\n";  
 }  
}

package Coaches;  
  
public class BussinesCoach extends Coach {  
 private int comfLevel;  
  
 public BussinesCoach(){  
 super("Бізнес клас",20,10,2000);  
 setComfLevel(3);  
 }  
  
 @Override  
 public int getComfLevel() {return comfLevel;}  
  
 public void setComfLevel(int comfLevel) {this.comfLevel = comfLevel;}  
  
 @Override  
 public String toString() {  
 return super.toString() +  
 " Рівень комфортності " + comfLevel + "\n";  
 }  
}

package Menu;  
  
import Train.Train;  
import java.util.\*;  
  
public class CountNumOfPassengersCommand implements Command {  
 private ArrayList<Train> trains;  
 private Scanner scan;  
  
 public CountNumOfPassengersCommand(ArrayList<Train> trains, Scanner scan) {  
 this.trains = trains;  
 this.scan = scan;  
 }  
  
 @Override  
 public void execute() {  
 System.*out*.println("Оберіть потрібний потяг:");  
 System.*out*.println("1 - Зимноводівський експрес");  
 System.*out*.println("2 - Львівський експрес");  
 System.*out*.println("3 - Солонківський поїзд");  
 System.*out*.println("4 - Рівненська електричка");  
 System.*out*.println("5 - Х'юндай");  
 System.*out*.println("6 - Лондон - Зимна Вода");  
 int choice = scan.nextInt() - 1;  
 if (choice >= 0 && choice < trains.size()) {  
 int numOfPass = trains.get(choice).numofpass();  
 System.*out*.println("У цьому потязі є " + numOfPass + " місць");  
 } else {  
 System.*out*.println("Неправильний вибір!");  
 }  
 }  
}

package Menu;  
  
import Train.Train;  
import java.util.ArrayList;  
import java.util.Scanner;  
  
public class CountNumOfBaggageCommand implements Command {  
 private ArrayList<Train> trains;  
 private Scanner scan;  
  
 public CountNumOfBaggageCommand(ArrayList<Train> trains, Scanner scan) {  
 this.trains = trains;  
 this.scan = scan;  
 }  
  
 @Override  
 public void execute() {  
 System.*out*.println("Оберіть потрібний потяг:");  
 System.*out*.println("1 - Зимноводівський експрес");  
 System.*out*.println("2 - Львівський експрес");  
 System.*out*.println("3 - Солонківський поїзд");  
 System.*out*.println("4 - Рівненська електричка");  
 System.*out*.println("5 - Х'юндай");  
 System.*out*.println("6 - Лондон - Зимна Вода");  
 int choice = scan.nextInt() - 1;  
 if (choice >= 0 && choice < trains.size()) {  
 int numOfBaggage = trains.get(choice).numofbagage();  
 System.*out*.println("У цьому потязі є " + numOfBaggage + " місць для багажу");  
 } else {  
 System.*out*.println("Неправильний вибір!");  
 }  
 }  
}

package Menu;  
  
import Train.Train;  
import java.util.ArrayList;  
import java.util.Scanner;  
  
public class SortCoachByComfortCommand implements Command {  
 private ArrayList<Train> trains;  
 private Scanner scan;  
  
 public SortCoachByComfortCommand(ArrayList<Train> trains, Scanner scan) {  
 this.trains = trains;  
 this.scan = scan;  
 }  
  
 @Override  
 public void execute() {  
 System.*out*.println("Оберіть потрібний потяг:");  
 System.*out*.println("1 - Зимноводівський експрес");  
 System.*out*.println("2 - Львівський експрес");  
 System.*out*.println("3 - Солонківський поїзд");  
 System.*out*.println("4 - Рівненська електричка");  
 System.*out*.println("5 - Х'юндай");  
 System.*out*.println("6 - Лондон - Зимна Вода");  
 int choice = scan.nextInt() - 1;  
 if (choice >= 0 && choice < trains.size()) {  
 trains.get(choice).sortCoaches();  
 System.*out*.println(trains.get(choice));  
 } else {  
 System.*out*.println("Неправильний вибір!");  
 }  
 }  
}

package Menu;  
  
import Train.Train;  
import java.util.ArrayList;  
import java.util.Scanner;  
  
public class FindCoachByPassCommand implements Command {  
 private ArrayList<Train> trains;  
 private Scanner scan;  
  
 public FindCoachByPassCommand(ArrayList<Train> trains, Scanner scan) {  
 this.trains = trains;  
 this.scan = scan;  
 }  
  
 @Override  
 public void execute() {  
 System.*out*.println("Оберіть потрібний потяг:");  
 System.*out*.println("1 - Зимноводівський експрес");  
 System.*out*.println("2 - Львівський експрес");  
 System.*out*.println("3 - Солонківський поїзд");  
 System.*out*.println("4 - Рівненська електричка");  
 System.*out*.println("5 - Х'юндай");  
 System.*out*.println("6 - Лондон - Зимна Вода");  
 int choice = scan.nextInt() - 1;  
 if (choice >= 0 && choice < trains.size()) {  
 System.*out*.println("Введіть кількість місць:");  
 int numOfPlaces = scan.nextInt();  
 trains.get(choice).findCoach(numOfPlaces);  
 } else {  
 System.*out*.println("Неправильний вибір!");  
 }  
 }  
}

**Git репозиторій:** https://github.com/zhyvoronkoRoman/LPNU\_PP\_Lab\_4-8

**Висновок**

На цій лабораторній роботі я закріпив можливості ООП: класи, успадкування, поліморфізм, інкапсуляцію