UNIVERSITY OF WESTMINSTER

INFORMATICS INSTITUTE OF TECHNOLOGY

Informatics Institute of Technology

School of Computing

Software Development II Coursework Report

Module : 4COSC010C.2: Software Development II (2023)

Date of submission : 25th March 2024

Student ID : <20230183> / <w2053190>

Student First Name : Misal

Student Surname : Silva

Tutorial group (day, time, and tutor/s): Group 21(Wednesday, 10.30AM, Ms.Vishmi Embuldeniya)

"I confirm that I understand what plagiarism / collusion / contract cheating is and have read and understood the section on Assessment Offences in the Essential Information for Students. The work that I have submitted is entirely my own. Any work from other authors is duly referenced and acknowledged."

Name : Misal Silva

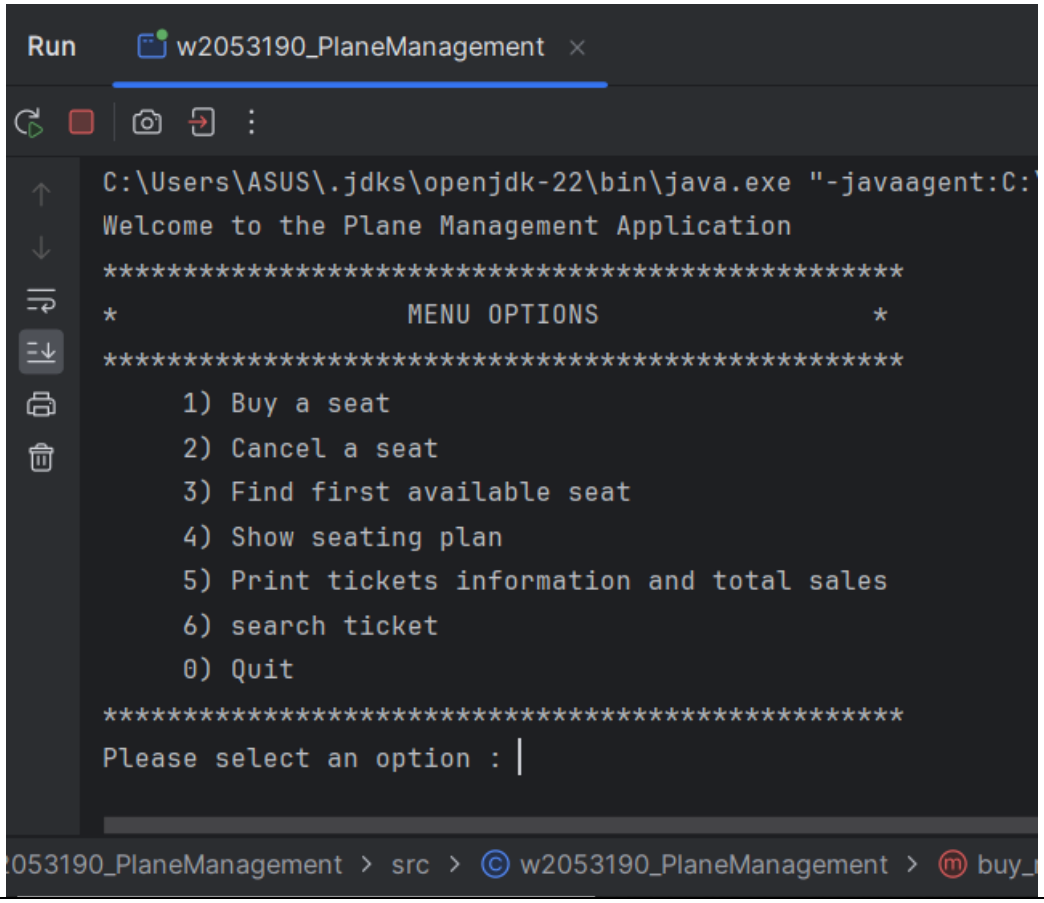
Student ID : <20230183> / <w2053190>

Self-assessment form and test plan

1) Self-assessment form

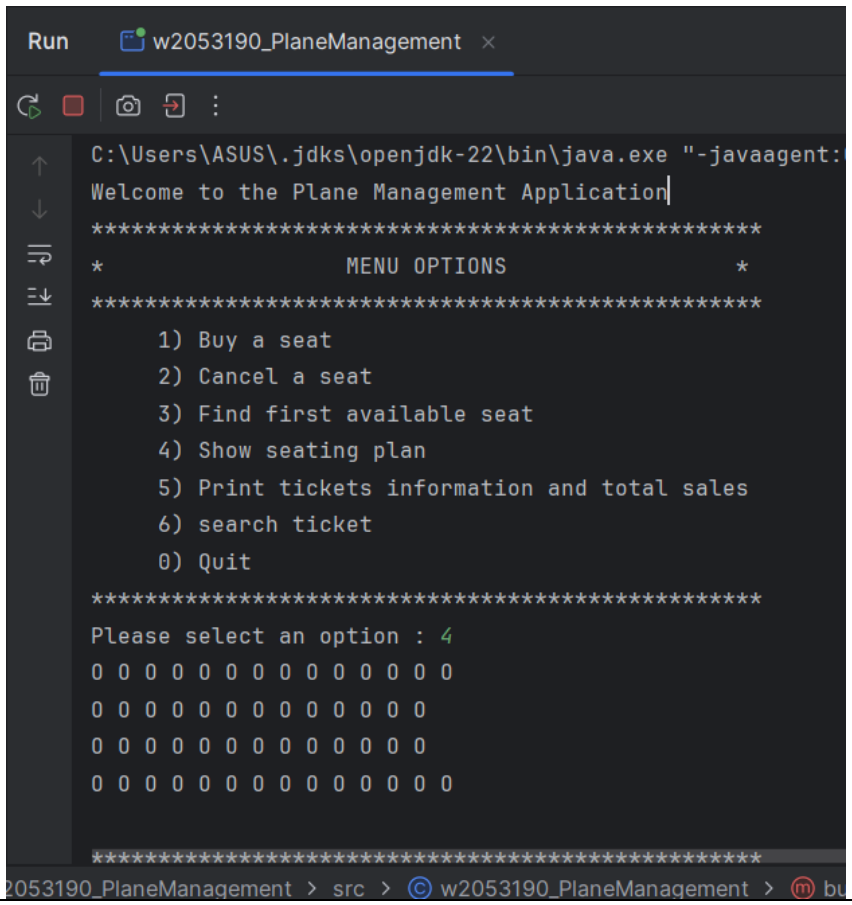
Task	Self-assessment (select one)	Comments
1	☒Fully implemented ☐Partially implemented ☐Not attempted	Prints the welcome message. Creates a 2D array for rows and columns. At the beginning 0 is assigned to all the seats.
2	☒Fully implemented ☐Partially implemented ☐Not attempted	Prints the menu options and prompts user to input by using switch-case control structure. Validates the input by try catch.

Insert here a screenshot of your welcome message and menu:



3	☒Fully implemented ☐Partially implemented ☐Not attempted	Prompts the user to input row number and seat number to buy a seat. Marks the seat as bought and assigns 1 to it.
4	☒Fully implemented ☐Partially implemented ☐Not attempted	Prompts the user to input row number and seat number to cancel a seat. Marks the seat as available and assigns 0 to it.
5	☒Fully implemented ☐Partially implemented ☐Not attempted	Creates 2 arrays for seat and row. Finds the first available seat using a nested for loop and breaks from them once a seat is found.
6	☒Fully implemented ☐Partially implemented ☐Not attempted	Displays the availability of seats using 0 and X. Uses a nested for loop to go through the 2D array.

Insert here a screenshot of the seating plan:



7	☒Fully implemented ☐Partially implemented ☐Not attempted	Person class is created by the given attributes. Constructors, getters and setters also created.
8	☒Fully implemented ☐Partially implemented ☐Not attempted	Ticket class is created by the given attributes. Constructors, getter, setters also created
9	☒Fully implemented ☐Partially implemented ☐Not attempted	Created a array of tickets. When buying seats it prompts the user to enter person information and stores them in the array. And when cancelling seats remove the tickets from the array .
10	☐Fully implemented ☒Partially implemented ☐Not attempted	Method print_ticket_info is called when the user enter '5', and it prints ticket information of booked seats and total sales of the sold tickets.
11	☒Fully implemented ☐Partially implemented ☐Not attempted	Method search_ticket is called when the user enter '6', and it search the seat given by the user and prints the information if it is occupied. Otherwise it will print that the seat is available
12	☒Fully implemented ☐Partially implemented ☐Not attempted	Save method save ticket information in to a text file when the user successfully buy a seat

2) Test Plan

Complete the test plan describing which testing you have performed on your program.
Add as many rows as you need.

Part A Testing

Test case / scenario	Input	Expected Output	Output	Pass/Fail
Task 1	Run main method.	The welcome message should be printed.	Welcome to the Plane Management application	☒Pass ☐Fail
Program Termination	option = 0	The menu should exit.	Exit.	☒Pass ☐Fail
Invalid menu option	Option = h	Invalid option / error message.	Invalid option. Please try again.	☒Pass ☐Fail
Buy seat	1.Option = 1 Row = A Seat = 1 2. Option = 1 Row = 2	1.Seat booked successfully. 2. Invalid option error message.	1.Seat A1 successfully bought. 2. Invalid input, enter between A,B,C and D	☒Pass ☐Fail
Cancel seat	1.Option = 2 Row = A Seat = 1 2. Option = 2 Row = B Seat = 1	1.Seat cancelled successfully. 2. seat is still available	1.Seat A1 cancelled successfully. 2. seat B1 is already available.	☒Pass ☐Fail
Find first available seat	Option = 3	if A1 is booked First available seat is A2	First available seat is A1	☒Pass ☐Fail
Show seat plan	Option = 4	Print available seats with ‘O’ and sold seats with ‘X	Print available seats with ‘O’ and sold seats with ‘X’	☒Pass ☒Fail

Part B testing

Test case / scenario	Input	Expected Output	Output	Pass/Fail
Buy a seat	Option = 1 Row = B Seat = 10	Display the prompt to get the user information and	Display the prompt to get the user information and	☒Pass ☐Fail

		add it into the array.	add it into the array.	
Cancel seat	Option = 2 Row = B Seat = 10	Seat cancelled. Information get removed from the array.	Seat cancelled. Information get removed from the array.	☒Pass ☐Fail
Print ticket info	Option = 5	Print all information of booked tickets and display total sales.	Print all information of booked tickets and display total sales.	☐Pass ☒Fail
Search ticket	Option = 6 Row = C Seat = 12	Print the information of the ticket and the status.	Print the information of the ticket and the status.	☒Pass ☐Fail
Save	Option = 1 Row = B Seat = 6 Name = Misal Surname = Silva Email = misalsilva@gmail.com	Save information of each seat in separate text file when the seat booked successfully.	Print all information of booked tickets and display total sales.	☒Pass ☐Fail
				☐Pass ☐Fail
				☐Pass ☐Fail

Are there any specific parts of the coursework which you would like to get feedback?

You will need to demonstrate your understanding of the submitted code. Your tutor will arrange a coursework demonstration. During the coursework demonstration, your tutor will ask you to execute your program and questions on your code.

Failure to attend the demonstration will result in 0 for the coursework.

3) Code :

W2053190_PlaneManagement.java

```
import java.util.InputMismatchException;
   import java.util.Scanner;
   public class w2053190_PlaneManagement {
      static Scanner input = new Scanner(System.in);
      //Define 2D Array to represent the seating plan of the plane
      static int[][] seat_plan = {
            {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,},
            {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,},
            {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,},
            {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,}
      };
      static int option;
      static boolean correct = false;
      static int max_seat_num;
      static double price;

      //Define an array to store ticket information
      static Ticket[] tickets = new Ticket[100];
      static int ticketCount = 0;

      //Method to display menu options and get user input for option
      static void menu_options(){
         correct = false;
         while(!correct){
            try{
               for (int i = 0; i < 50; i++) {
                  System.out.print("*");
               }
```

```
            System.out.println();
            System.out.println("*                MENU OPTIONS                *");
            for (int i = 0; i < 50; i++) {
               System.out.print("*");
            }
            System.out.println();
            System.out.println("    1) Buy a seat");
            System.out.println("    2) Cancel a seat");
            System.out.println("    3) Find first available seat");
            System.out.println("    4) Show seating plan");
            System.out.println("    5) Print tickets information and total sales");
            System.out.println("    6) search ticket");
            System.out.println("    0) Quit");
            for (int i = 0; i < 50; i++) {
               System.out.print("*");
            }
            System.out.println();
            System.out.print("Please select an option : ");
            option  = input.nextInt();
            //set correct to true if input is valid
            correct = true;
         }
         catch(InputMismatchException e){
            //handling invalid input
            System.out.println("Invalid Input");
            input.nextLine();
            correct = false;
         }
      }
   }
   //Method to buy a seat
   static void buy_method() {
```

```
char row_letter = 0;
int seat_num = 0;
boolean correct = false;

while (!correct) {
   try {
      //prompting for row letter
      System.out.print("Enter Row Letter: ");
      //using toUpperCase so that simple characters converts to capital
      row_letter = input.next().toUpperCase().charAt(0);
      //checks if the row_letter entered by the user is within the valid range of seat rows
      if (row_letter < 'A' || row_letter > 'D') {
         System.out.println("Invalid Row Letter. Please enter A, B, C, or D.");
         buy_method();
      }
      correct = true;
   }
   catch (InputMismatchException e) {
      //handling invalid input
      System.out.println("Invalid Row Letter, please try again");
      input.nextLine();
   }

   try {
      System.out.print("Enter Seat Number: ");
      seat_num = input.nextInt();
      int max_seat_num;
      //assigning values to max_seat_num according to the row letter
      if (row_letter == 'B' || row_letter == 'C')
         max_seat_num =  12;
      else {
         max_seat_num = 14;
```

```
            }
            //checking seat_num if within valid range of seat numbers
            if (seat_num >= 1 && seat_num <= max_seat_num){
               correct = true;
            }
            else {
               System.out.println("Invalid seat number. Please enter between 1 and " +
max_seat_num + ".");
               input.nextLine();
               continue;
            }
            correct = true;
         }
         catch (InputMismatchException e) {
            System.out.println("Invalid Seat Number, please try again");
            input.nextLine();
         }
      }
      //prompting for person information
      input.nextLine();
      System.out.print("Enter person's name: ");
      String name = input.nextLine();
      System.out.println();
      System.out.print("Enter person's surname: ");
      String surname = input.nextLine();
      System.out.print("Enter person's email: ");
      String email = input.nextLine();
      //new person object
      Person person = new Person(name, surname, email);
       //assign price values according to the seats
      if (seat_num >= 1 && seat_num <= 5){
         double price = 200.0;
```

```
    }
    else if (seat_num >= 6 && seat_num <= 9){
        double price = 150.0;
    }
    else if (seat_num >= 10 && seat_num <= max_seat_num){
        double price = 180.0;
    }
    //new ticket object
    Ticket ticket = new Ticket(String.valueOf(row_letter), seat_num, price, person);
     tickets[ticketCount] = ticket;
     ticketCount++;


    int row_index = row_letter - 'A';
    int seat_index = seat_num - 1;

    if (seat_plan[row_index][seat_index] == 1) {
        System.out.println("Seat " + row_letter + seat_num + " is already occupied.");
    }
    else {
        seat_plan[row_index][seat_index] = 1;
        System.out.println("Seat " + row_letter + seat_num + " bought Successfully.");
    }
    //calling save method in the ticket class
    ticket.save();
}
//Method to cancel a seat
static void cancel_seat() {
    char row_letter = 0;
    int seat_num = 0;
    boolean correct = false;
```

```
while (!correct) {
    try {
        //prompting for row letter
        System.out.print("Enter Row Letter: ");
        //using toUpperCase so that simple characters converts to capital
        row_letter = input.next().toUpperCase().charAt(0);
        if (row_letter < 'A' || row_letter > 'D') {
            //checks if the row_letter entered by the user is within the valid range of seat rows
            System.out.println("Invalid Row Letter. Please enter A, B, C, or D.");
            return;
        }
        correct = true;
    }
    catch (InputMismatchException e) {
        //handling invalid input
        System.out.println("Invalid Row Letter, please try again");
        input.nextLine();
    }

    try {
        System.out.print("Enter Seat Number: ");
        seat_num = input.nextInt();
        int max_seat_num;
        //assigning values to max_seat_num according to the row letter
        if (row_letter == 'B' || row_letter == 'C')
            max_seat_num =  12;
        else {
            max_seat_num = 14;
        }
        //checking seat_num if within valid range of seat numbers
        if (seat_num >= 1 && seat_num <= max_seat_num){
            correct = true;
```

```
            }
            else {
                System.out.println("Invalid seat number. Please enter between 1 and " +
max_seat_num + ".");
                return;
            }
            correct = true;
        }
        catch (InputMismatchException e) {
            //handling invalid input
            System.out.println("Invalid Seat Number, please try again");
            input.nextLine();
        }
    }
    boolean found = false;
    for (int i =0; i < ticketCount; i++){
        //locating the row and seat
        Ticket ticket = tickets[i];
        if (ticket.getRow().charAt(0) == row_letter && ticket.getSeat() == seat_num){
            for (int j = 0; j < ticketCount - 1; j++){
                tickets[j] = tickets[j + 1];
            }
            //once located it will be removed from the array
            ticketCount--;
            found = true;

            int row_index = row_letter - 'A';
            int seat_index = seat_num - 1;

            if (seat_plan[row_index][seat_index] == 0) {
                System.out.println("Seat " + row_letter + seat_num + " is already available.");
            }
```

```
            else {
                seat_plan[row_index][seat_index] = 0;
                System.out.println("Seat " + row_letter + seat_num + " cancelled Successfully.");
            }
        }
        if (!found){
            //if not located error message will be rinted
            System.out.println("Tickets not found for seat " + row_letter + seat_num);
        }
    }
}
//method to find the first available seat
static void find_first_available(){
    String[] rows = {"A", "B", "C", "D"};
    int[] seats = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14};
    boolean found = false;

    //goes through the whole array and checks the first available seat
    for (int i = 0; i<4; i++){
        for (int j = 0; j< seats.length; j++){
            if (seat_plan[i][j] == 0){
                System.out.println("The first available seat is "+rows[i]+seats[j]);
                found = true;
                break;
            }
        }
        //once the available seat is found will break form the loop
        if (found){
            break;
        }
    }
    //if not found error message will be printed
```

```
    if (!found){
        System.out.println("There are no seats available.");
    }
}
//method to print the seating plan
static void show_seating_plan(){
    for (int i = 0; i<4; i++){
        for (int j = 0;j < seat_plan[i].length; j++){
            if (seat_plan[i][j] == 0){
                System.out.print("O ");
            }
            else if(seat_plan[i][j] == 1){
                System.out.print("X ");
            }
        }
        System.out.println();
    }
    System.out.println();
}

//method to print ticket information
static void print_tickets_info(){
    double totalAmount = 0.0;

    System.out.println("Tickets sold during the session: ");
    for (int i = 0; i < ticketCount; i++){
        Ticket ticket = tickets[i];
        System.out.println("Ticket " + (i + 1) + ".");
        ticket.printTicketInfo();
        totalAmount += ticket.getPrice();
    }
    System.out.println("Total amount: £" + totalAmount);
```

```
}
//method to search tickets
static void search_tickets(){
    char row_letter = 0;
    int seat_num = 0;
    boolean correct = false;

    while (!correct) {
        try {
            System.out.print("Enter Row Letter: ");
            row_letter = input.next().toUpperCase().charAt(0);
            if (row_letter < 'A' || row_letter > 'D') {
                System.out.println("Invalid Row Letter. Please enter A, B, C, or D.");
                return;
            }
            correct = true;
        }
        catch (InputMismatchException e) {
            System.out.println("Invalid Row Letter, please try again");
            input.nextLine();
        }

        try {
            System.out.print("Enter Seat Number: ");
            seat_num = input.nextInt();
            int max_seat_num;
            if (row_letter == 'B' || row_letter == 'C')
                max_seat_num =  13;
            else {
                max_seat_num = 14;
            }
            if (seat_num >= 1 && seat_num <= max_seat_num){
```

```
                correct = true;
            }
            else {
                System.out.println("Invalid seat number. Please enter between 1 and " +
max_seat_num + ".");
                return;
            }
            correct = true;
        }
        catch (InputMismatchException e) {
            System.out.println("Invalid Seat Number, please try again");
            input.nextLine();
        }
      }
      boolean found = false;
      for (int i =0; i < ticketCount; i++) {
        Ticket ticket = tickets[i];
        if (ticket.getRow().charAt(0) == row_letter && ticket.getSeat() == seat_num) {
            System.out.println("Ticket Information: ");
            ticket.printTicketInfo();
            found = true;
            break;
        }
      }
      if (!found){
        System.out.println("This seat is available.");
      }
    }

    public static void main(String[]args){
      System.out.println("Welcome to the Plane Management Application");
      while(true){
```

```
        menu_options();

        switch(option){
            case 1: buy_method(); break;
            case 2: cancel_seat(); break;
            case 3: find_first_available(); break;
            case 4: show_seating_plan(); break;
            case 5: print_tickets_info(); break;
            case 6: search_tickets(); break;
            case 0: System.out.println("Exiting..."); return;
            default: System.out.println("Invalid Option. Please try again."); break;
            }
        }
    }
}
```

Person.java

```
public class Person {
//Attributes
private String name;
private String surname;
private String email;

//constructor
public Person(String name, String surname, String email){
    this.name = name;
    this.surname = surname;
    this.email = email;
}

//Getter and setter for name
public String getName(){
```

```
        return name;
    }
    public void setName(String name){
        this.name = name;
    }
    //Getter and setter for surname
    public String getSurname(){
        return surname;
    }
    public void setSurname(String surname){
        this.surname = surname;
    }
    //Getter and setter for email
    public String getEmail(){
        return email;
    }
    public void setEmail(String email){
        this.email = email;
    }
    //method for printing person information
    public void printPersonInfo(){
        System.out.println("Name is " + name);
        System.out.println("Surname is " + surname);
        System.out.println("Email is " + email);
    }
}
```

Ticket.java

```
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
public class Ticket {
   //Attributes
   private String row;
   private int seat;
   private double price;
   private Person person;

   //constructor
   public Ticket(String row, int seat, double price, Person person) {
      this.row = row;
      this.seat = seat;
      this.price = price;
      this.person = person;
   }

   //Getter and setter for row
   public String getRow() {
      return row;
   }

   public void setRow(char row) {
      this.row = String.valueOf(row);
   }

   //getter and setter for seat
   public int getSeat() {
      return seat;
```

```
}

public void setSeat(int seat) {
   this.seat = seat;
}

//getter and setter for price
public double getPrice() {
   return price;
}

public void setPrice(double price) {
   this.price = price;
}

//getter and setter for person
public Person getPerson() {
   return person;
}

public void setPerson(Person person) {
   this.person = person;
}

//method for printing seat information and person information
public void printTicketInfo() {
   System.out.println("Row is " + row);
   System.out.println("Seat is " + seat);
   System.out.println("Price is " + price);
   System.out.println("Person information is: ");
   person.printPersonInfo();
}
```

```
    //method for saving ticket and person information when buying a seat
    public void save() {
        String fileName = this.getRow() + this.getSeat() + ".txt";
        try (FileWriter writer = new FileWriter(row + "" + seat + ".txt")) {
            writer.write("Ticket Information of seats" + "\n");
            writer.write("\t Row " + getRow() + "\n");
            writer.write("\t Seat " + getSeat() + "\n");
            writer.write("\t Price £ " + getPrice() + "\n");
            writer.write("\t Person Information: " + "\n");
            writer.write("\t First name: " + person.getName() +
                    "\n");
            writer.write("\t Surname: " + person.getSurname() +
                    "\n");
            writer.write("\t Email: " + person.getEmail() + "\n");
            writer.close();
        } catch (IOException e) {
            e.printStackTrace();
        }

    }
}
```

<<END>>