Informatics Institute of Technology

School of Computing

Software Development II Coursework Report

Module	: 4COSC010C.2: Software Development II (2023)
Date of submission	: 23/03/2024
Student ID	: 20231611 / w2053073
Student First Name	: Abeysing
Student Surname	: Rasanjana

Tutorial group (day, time, and tutor/s): Group 8

"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."

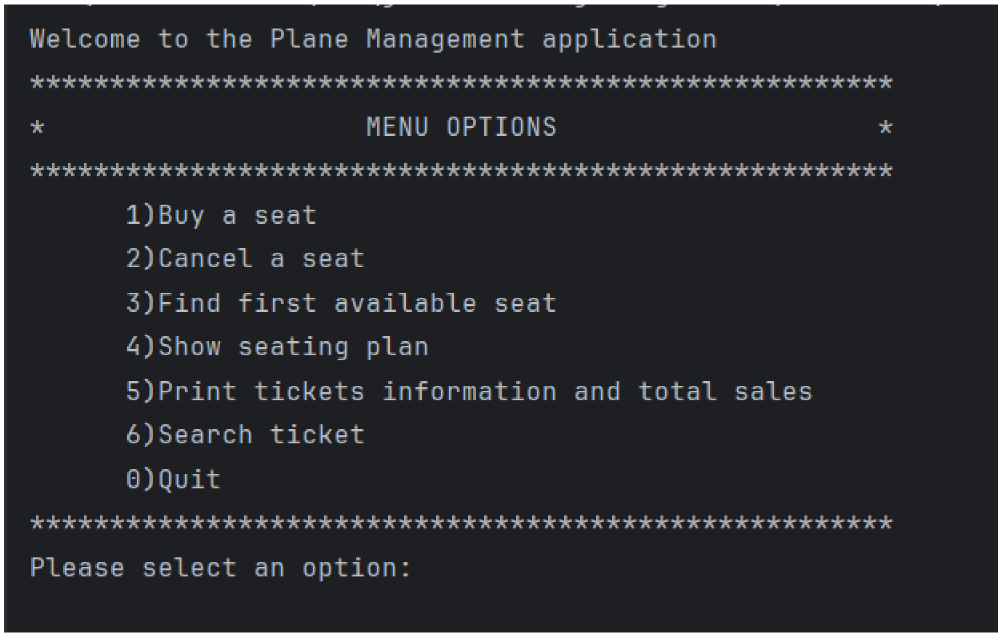
Name	: Pinidu Rasanjana
Student ID	: 20231611

Self-assessment form and test plan

1) Self-assessment form

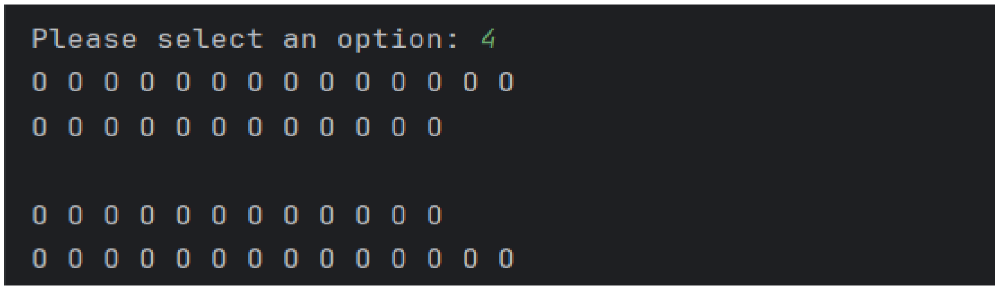
Task	Self-assessment (select one)	Comments
1	☒Fully implemented ☐Partially implemented ☐Not attempted	
2	☒Fully implemented ☐Partially implemented ☐Not attempted	

Insert here a screenshot of your welcome message and menu:



```
Welcome to the Plane Management application
******************************************************
*                    MENU OPTIONS                    *
******************************************************
      1)Buy a seat
      2)Cancel a seat
      3)Find first available seat
      4)Show seating plan
      5)Print tickets information and total sales
      6)Search ticket
      0)Quit
******************************************************
Please select an option:
```

Task	Self-assessment (select one)	Comments
3	☒Fully implemented ☐Partially implemented ☐Not attempted	
4	☒Fully implemented ☐Partially implemented ☐Not attempted	
5	☒Fully implemented ☐Partially implemented ☐Not attempted	
6	☒Fully implemented	

	☐Partially implemented ☐Not attempted	
Insert here a screenshot of the seating plan: 		

```
Please select an option: 4
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
```

7	☒Fully implemented ☐Partially implemented ☐Not attempted	
8	☒Fully implemented ☐Partially implemented ☐Not attempted	
9	☒Fully implemented ☐Partially implemented ☐Not attempted	
10	☒Fully implemented ☐Partially implemented ☐Not attempted	
11	☒Fully implemented ☐Partially implemented ☐Not attempted	
12	☒Fully implemented ☐Partially implemented ☐Not attempted	

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**
buy_seat() method	Choose option 1 (Buy a seat), provide name, surname, email, row	Check for seat is available or not. if it's , Confirmation message that	All tasks carried out well.	☒Pass ☐Fail

	letter, and seat number.	the ticket is purchased successfully. If It’s not, show that seat is sold.		
cancel_seat() method	Choose option 2 (Cancel a seat), provide row and seat number.	Check whether that seat is already available or not, If it’s available , cancel that seat and show message cancelled.	Cancels a seat.	☒Pass ☐Fail
find_first_available() method	Choose option 3 (Find first available seat).	Information about the first available seat.	Executes Properly.	☒Pass ☐Fail
show_seating_plan() method	Choose option 4 (Show seating plan).	Display the seating plan showing available(O) and occupied(X) seats.	Executes Properly.	☒Pass ☐Fail
Quit	Choose option 0.	Quit the main programme.	Quit	☒Pass ☐Fail
				☐Pass ☐Fail
				☐Pass ☐Fail

Part B testing

Test case / scenario	**Input**	**Expected Output**	**Output**	**Pass/Fail**
Person class	Name,SurName,email	All variables have accessible getters and setters.	Each variable's getters and setters conform to the expected output.	☒Pass ☐Fail

Ticket class	Row, seat, price and person_info	Constructor invocation from another class allows passing values for parameters, facilitating the creation of objects	The constructor can be invoked from another class, allowing the passing of values for parameters to create objects as expected.	☒Pass ☐Fail
save() method		Generates a file to store seat allocations, including rows and names, and proceeds to write the relevant information for each person.		☒Pass ☐Fail
print_tickets_info() method		Procuring seats randomly across assorted price tiers, we examine whether this approach successfully prints the seat number, row, person information and associated price.	Functions correctly.	☒Pass ☐Fail
search_ticket() method	Seat number and Row letter.	Inputting a reserved seat row and number, it verifies	Functions correctly.	☒Pass ☐Fail

		whether the system returns the customer's information.		

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 <u>0 for the coursework.</u>

3) Code :

Plane Management Class:

```
import java.util.Arrays;

import java.util.Scanner;

import java.util.InputMismatchException;

public class w2053073_PlaneManagement {

    static int[][] seats = new int[5][];

    static Ticket[][] tickets = new Ticket[5][];

    public static void main(String[] args) {

        Scanner input = new Scanner(System.in);

        seats[0] = new int[14];

        seats[1] = new int[12];

        seats[2] = new int[0];

        seats[3] = new int[12];

        seats[4] = new int[14];

        tickets[0] = new Ticket[14];

        tickets[1] = new Ticket[12];

        tickets[2] = new Ticket[0];

        tickets[3] = new Ticket[12];

        tickets[4] = new Ticket[14];
while (true){
    System.out.println("Welcome to the Plane Management application");
    String asterisks = "*".repeat(54);
    String space =  " ".repeat(20);
    System.out.println(asterisks);
    System.out.println("*"+space+"MENU OPTIONS"+space+"*");
    System.out.println(asterisks);
    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              ");
    System.out.println(asterisks);
    System.out.print("Please select an option: ");
    try {
        int option = input.nextInt();
        switch (option) {
            case 1:
                buy_seat();
    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_ticket();
    break;
case 0:
    return;
default:
    System.out.println("Please enter a valid option.");
    break;
```

```
            }
        } catch (InputMismatchException e) {
            System.out.println("Invalid input. Please enter a valid number for option.");
            input.nextLine();
            continue;
        }
    }
}
public static void buy_seat() {
    int row_int = 0;
    Scanner input = new Scanner(System.in);
    System.out.print("Please enter your name: ");
    String name = input.next();
    System.out.print("Please enter your surname: ");
    String surname = input.next();
    System.out.print("Please enter your email: ");
    String email = input.next();
    System.out.print("Input the row letter: ");
    String rowforbuy = input.next().toUpperCase();
    System.out.print("Input the seat number: ");
    int numforbuy = input.nextInt();
```

```
if (rowforbuy.equals("A")) {

    row_int = 0;

} else if (rowforbuy.equals("B")) {

    row_int = 1;

} else if (rowforbuy.equals("C")) {

    row_int = 3;

} else if (rowforbuy.equals("D")) {

    row_int = 4;

}

if (!Arrays.asList("A", "B", "C", "D").contains(rowforbuy)) {

    System.out.println("Invalid row entered.");

    return;

}

if (seats[row_int][numforbuy - 1] != 1) {

    seats[row_int][numforbuy - 1] = 1;

    if (numforbuy <= 5) {

        Person newperson = new Person(name, surname, email);

        Ticket newticket = new Ticket(rowforbuy, numforbuy, 200, newperson);

        tickets[row_int][numforbuy - 1] = newticket;

        newticket.save();

    } else if (numforbuy >= 10) {
            Person newperson = new Person(name, surname, email);

            Ticket newticket = new Ticket(rowforbuy, numforbuy, 180, newperson);

            tickets[row_int][numforbuy - 1] = newticket;

            newticket.save();

        } else {

            Person newperson = new Person(name, surname, email);

            Ticket newticket = new Ticket(rowforbuy, numforbuy, 150, newperson);

            tickets[row_int][numforbuy - 1] = newticket;

            newticket.save();

        }

        System.out.println("Ticket purchased successfully!");

    } else {

        System.out.println("This seat is sold..");

    }

}

public static void cancel_seat(){

    Scanner input = new Scanner(System.in);

    System.out.print("Input the row letter: ");

    String rowforcancel = input.next().toUpperCase();

    System.out.print("Input the seat number: ");

    int numforcancel = input.nextInt();
int row_int = 0;

if (rowforcancel.equals("A")) {

    row_int = 0;

} else if (rowforcancel.equals("B")) {

    row_int = 1;

} else if (rowforcancel.equals("C")) {

    row_int = 3;

} else if (rowforcancel.equals("D")) {

    row_int = 4;

}

if (!Arrays.asList("A", "B", "C", "D").contains(rowforcancel)) {

    System.out.println("Invalid row entered.");

    return;

}

if (seats[row_int][numforcancel - 1] == 1) {

    seats[row_int][numforcancel - 1] = 0;

    tickets[row_int][numforcancel - 1] = null;

}else {

    System.out.println("That seat is not available already...");

}

System.out.println("Seat is canceled");
```

```
}

public static void find_first_available(){

    int row = 0;

    int col = 0;

    while (row < seats.length) {

        if (seats[row][col] == 0) {

            if (row == 0) {

                System.out.println("Seat is still available: " + "A" + (col + 1));

            } else if (row == 1) {

                System.out.println("Seat is still available: " + "B" + (col + 1));

            } else if (row == 2) {

                System.out.println("Seat is still available: " + "C" + (col + 1));

            } else {

                System.out.println("Seat is still available: " + "D" + (col + 1));

            }

            break;

        }

        col++;

        if (col >= seats[row].length) {

            row++;

            col = 0;
        }
    }
}
public static void show_seating_plan(){
    for (int i = 0; i < seats.length; i++) {
        for (int j = 0; j < seats[i].length; j++) {
            if (seats[i][j] == 0) {
                System.out.print("O ");
            } else {
                System.out.print("X ");
            }
        }
        System.out.println();
    }
}
public static void print_tickets_info(){
    System.out.println("********* Ticket Information *********");
    int total = 0;
    for (int i = 0; i < tickets.length; i++) {
        for (int j = 0;j <tickets[i].length;j++){
            Ticket ticket = tickets[i][j];
            if (ticket!=null) {
                System.out.println("Row: "+ticket.getRow());
                System.out.println("Seat Number: " + ticket.getSeat());
                System.out.println("Price: " + ticket.getPrice());
                System.out.println("Person Information:");
                System.out.println("Name: " + ticket.getPerson().getName());
                System.out.println("Surname: " + ticket.getPerson().getSurname());
                System.out.println("Email: " + ticket.getPerson().getEmail());
                System.out.println("--------------------------------------------------------");
                total += ticket.getPrice();
            }
        }
    }
    System.out.println("Total amount : £"+total);
}


public static void search_ticket() {
    Scanner input = new Scanner(System.in);
    System.out.print("Input the row letter: ");
    String rowforsearch = input.next();
    System.out.print("Input the seat number: ");
```

```
int seatforsearch = input.nextInt();

int row_int = 0;

if (!Arrays.asList("A", "B", "C", "D").contains(rowforsearch)) {

    System.out.println("Invalid row entered.");

    return;

}

if (rowforsearch.equals("A")){

    row_int = 0;

}else if (rowforsearch.equals("B")){

    row_int = 1;

}else if (rowforsearch.equals("C")) {

    row_int = 3;

}else if (rowforsearch.equals("D")) {

    row_int = 4;

}

Ticket ticket = tickets[row_int][seatforsearch-1];

if (tickets[row_int][seatforsearch-1] != null){

    System.out.println("Row: " + ticket.getRow());

    System.out.println("Seat Number: " + ticket.getSeat());

    System.out.println("Price: " + ticket.getPrice());

    System.out.println("Person Information:");
```

```
            System.out.println("Name: " + ticket.getPerson().getName());

            System.out.println("Surname: " + ticket.getPerson().getSurname());

            System.out.println("Email: " + ticket.getPerson().getEmail());

        }else {

            System.out.println("This seat is available..");

        }

    }

}
```

Ticket Class:

```
import java.io.FileWriter;

import java.io.IOException;

public class Ticket{

    private String row;

    private int seat;

    private double price;

    private Person person;

    public Ticket(String row, int seat, double price, Person person) {

        this.row = row;

        this.seat = seat;

        this.price = price;
    this.person = person;

}

public void setRow(String row){

    this.row = row;

}

public String getRow(){

    return row;

}

public void setSeat(int seat){

    this.seat = seat;

}

public int getSeat(){

    return seat;

}

public void setPrice(double price){

    this.price = price;

}

public double getPrice(){

    return price;

}

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

    }

    public Person getPerson(){

        return person;

    }

    public void save() {

        String fileName = row + seat + ".txt";

        try (FileWriter writer = new FileWriter(fileName)) {

            writer.write("Row: " + row + "\n");

            writer.write("Seat Number: " + seat + "\n");

            writer.write("Price: " + price + "\n");

            writer.write("Person Information:\n");

            writer.write("Name: " + person.getName() + "\n");

            writer.write("Surname: " + person.getSurname() + "\n");

            writer.write("Email: " + person.getEmail() + "\n");

            System.out.println("Ticket information saved to file: " + fileName);

        } catch (IOException e) {

            System.out.println("An error occurred while saving the ticket information to file.");

        }

    }

}
```

Person Class:

```
public class Person {

    private String name;

    private String surname;

    private String email;



    public Person(String name, String surname, String email){

        this.name = name;

        this.surname = surname;

        this.email = email;

    }

    public void setName(String name){

        this.name = name;

    }

    public String getName(){

        return name;

    }

    public void setSurname(String surname){

        this.surname = surname;

    }

    public String getSurname(){
        return surname;
    }
    public void setEmail(String email){
        this.email = email;
    }
    public String getEmail(){
        return email;
    }
}
```

<<END>>