

Informatics Institute of Technology
School of Computing
Software Development II Coursework Report

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

Date of submission : 03/03/2024

Student ID : 20230746 / w2051756

Student First Name : Agzaiyenth

Student Surname : Ganaraj

Tutorial group 3 (Monday, 10.30am, Mr. Ruwan):

"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 : Agzaiyenth Ganaraj

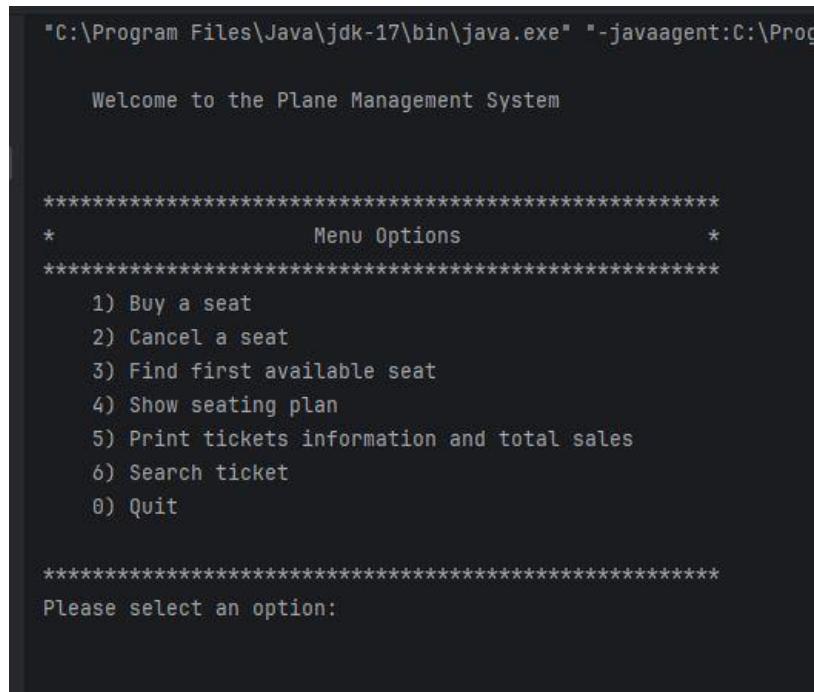
Student ID : 20230746/w2051756

Self-assessment form and test plan

1) Self-assessment form

Task	Self-assessment (select one)	Comments
1/	<input checked="" type="checkbox"/> Fully implemented <input type="checkbox"/> Partially implemented <input type="checkbox"/> Not attempted	
2	<input checked="" type="checkbox"/> Fully implemented <input type="checkbox"/> Partially implemented <input type="checkbox"/> Not attempted	

Insert here a screenshot of your welcome message and menu:



```
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Prog
Welcome to the Plane Management System

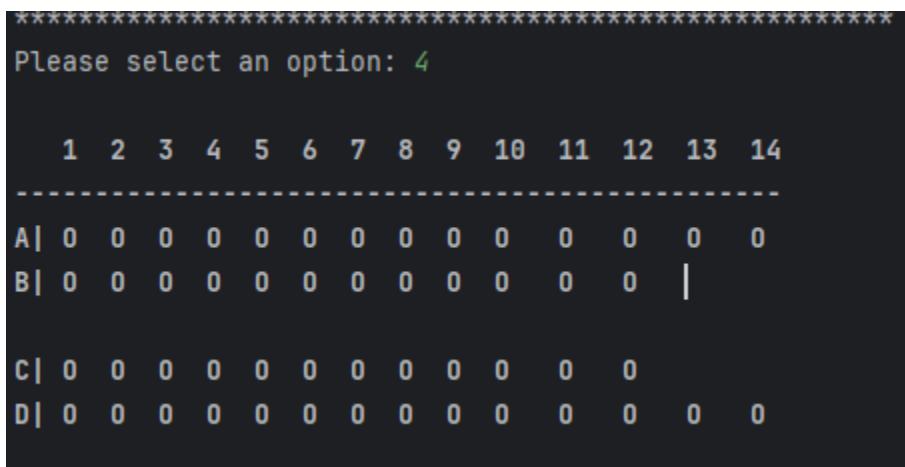
*****
*          Menu Options          *
*****
1) Buy a seat
2) Cancel a seat
3) Find first available seat
4) Show seating plan
5) Print tickets information and total sales
6) Search ticket
0) Quit

*****
Please select an option:
```

3	<input checked="" type="checkbox"/> Fully implemented <input type="checkbox"/> Partially implemented <input type="checkbox"/> Not attempted	
4	<input checked="" type="checkbox"/> Fully implemented <input type="checkbox"/> Partially implemented <input type="checkbox"/> Not attempted	
5	<input checked="" type="checkbox"/> Fully implemented	

	<input type="checkbox"/> Partially implemented <input type="checkbox"/> Not attempted	
6	<input checked="" type="checkbox"/> Fully implemented <input type="checkbox"/> Partially implemented <input type="checkbox"/> Not attempted	

Insert here a screenshot of the seating plan:



7	<input checked="" type="checkbox"/> Fully implemented <input type="checkbox"/> Partially implemented <input type="checkbox"/> Not attempted	
8	<input checked="" type="checkbox"/> Fully implemented <input type="checkbox"/> Partially implemented <input type="checkbox"/> Not attempted	
9	<input checked="" type="checkbox"/> Fully implemented <input type="checkbox"/> Partially implemented <input type="checkbox"/> Not attempted	
10	<input checked="" type="checkbox"/> Fully implemented <input type="checkbox"/> Partially implemented <input type="checkbox"/> Not attempted	
11	<input checked="" type="checkbox"/> Fully implemented <input type="checkbox"/> Partially implemented <input type="checkbox"/> Not attempted	
12	<input checked="" type="checkbox"/> Fully implemented <input type="checkbox"/> Partially implemented <input type="checkbox"/> 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
Task 1	N/A	Creating array with initial values as 0	Created an array with 0 as initial values	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail
Task 2	0	'User menu'	'User menu'	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail
Task 3	1, row, seat number	'Purchased a seat successfully'	'Purchased a seat successfully'	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail
Task 4	2, row, seat number	'Cancelled a seat successfully'	'Cancelled a seat successfully'	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail
Task 5	3	First seat Available	First seat Available	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail
Task 6	4	Shows seating plan	Shows seating plan	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail

Part B testing

Test case / scenario	Input	Expected Output	Output	Pass/Fail
Task 7	N/A	Creates a person object	Creates a Person Object	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail
Task 8	N/a	Creates a Ticket Object	Creates a Ticket Object	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail
Task 9	1	Creates an object out of person and ticket successfully	Creates an object out of person and ticket successfully	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail
Task 9	2	Cancels the seat and deletes the ticket object created	Cancels the seat and deletes the ticket object created	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail
Task 10	5	Prints Ticket	Prints Ticket	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail

Task 11	6	Search Ticket	Search Ticket	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> Fail
Task 12	1	Saves ticket as a text file	Saves ticket as a text file	<input checked="" type="checkbox"/> Pass <input type="checkbox"/> 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:

<<Code's pasted as text >>

<<PlaneManagement.java >>

```
import java.util.*;
public class PlaneManagement{
    //creating instance variables
    private static final Ticket [] tickets=new Ticket[52];
    static Scanner scanner = new Scanner(System.in);
    private static int ticketCount=0;
    private static final int[][] seats = new int[4][];
    //Methods
    public static int user_menu() {
        /*
            Displays User menu to the user
            @returns the selected option in 'int' data type
        */
        System.out.println("\n\n");
        System.out.print("*****");
        *                               Menu Options
        ****
        \t1) Buy a seat
        \t2) Cancel a seat
        \t3) Find first available seat
        \t4) Show seating plan
        \t5) Print tickets information and total sales
        \t6) Search ticket
        \t0) Quit

        *****");
        System.out.print("Please select an option: ");
        int response = 10;
        try {
            response = new Scanner(System.in).nextInt();
        } catch (InputMismatchException e) {
            //Take a break,do nothing!
        }
        System.out.println();
        return response;
    }
    public static void buy_seat() {
        /*

```

```

        Allows user to purchase seats
    */
    show_seating_plan();
    switcher(2);
}
public static void cancel_seat(){
/*
    Allows the user to cancel the seat which has been booked
previously
*/
    switcher(1);
}
public static void find_first_available(){
/*
    Prints the first seat row and number of the seat which is
available.
*/
String row_letter="";
breaker:
for(int rows=0;rows<seats.length;rows++) {
    for(int seatnumber=0;seatnumber<seats[rows].length;seatnumber++) {
        if(seats[rows][seatnumber]==0) {
            row_letter = switch_(rows) {
                case 0 -> "A";
                case 1 -> "B";
                case 2 -> "C";
                case 3 -> "D";
                default -> row_letter;
            };
            System.out.println("Seat in row "+row_letter+" with seat
number "+(seatnumber+1)+" is available");
            break breaker;
        }
    }
}
public static void show_seating_plan(){
/*
    Prints Seats that are available and the sold ones, 'O' denotes the
free seats & 'X' denotes the sold seats
*/
String[] Rows={"A","B","C","D"};
System.out.println("\u001b[1m   1   2   3   4   5   6   7   8   9   10  11  12
13  14");
System.out.println("-----");
for (int x=0;x<seats.length;x++) {
    if(x==2) System.out.println();
    System.out.print("\u001b[1m"+Rows[x]+" | ");
    for(int n=0;n<seats[x].length;n++) {
        if(seats[x][n]==0 && n>=9) {
            System.out.print("O   ");
        }else if(seats[x][n]==1 && n>=9) {
            System.out.print("X   ");
        }else if(seats[x][n]==0) {
            System.out.print("O   ");
        }else {

```

```

        System.out.print("X  ");
    }
}
System.out.println();
}
System.out.println();
}

public static void print_tickets_info() {
/*
    Prints all the Ticket information sold
*/
int totalAmount = 0;
System.out.print("\nTickets Sold During Session: \n");
for (int i = 0; i < ticketCount; i++) {
    System.out.println("\nTicket " + (i + 1) + ":");
    tickets[i].printInfo();
    totalAmount += (int) tickets[i].getPrice();
}
System.out.println("\nTotal Sales: £" +totalAmount);
}

public static void search_ticket() {
/*
    Checks if the seat is available or sold,if sold it prints the
user info
*/
switcher(3);
}

public static void specificTicketFinder(String rowLetter,int
rowNumber,int conditionController){
/*
 * check's for the specific ticket and prints the information
 * @params Row letter, row number & stop condition controller
 */
System.out.print("Enter the seat number: ");
try {
    int seat_number = scanner.nextInt();
    if(seat_number>0 && seat_number<conditionController){
        if (seats[rowNumber][seat_number - 1] == 1) {
            for (int i = 0; i < ticketCount; i++) {
                Ticket ticket = tickets[i];
                if (Objects.equals(ticket.getRow(),
rowLetter.toUpperCase()) && ticket.getSeat() == seat_number - 1) {
                    ticket.printInfo();
                    break;
                }
            }
        } else{
            System.out.println("Seat is not purchased yet");
        }
    } else{
        System.out.println("Enter a valid Seat number");
    }
} catch (Exception e) {
    System.out.println("Enter an Integer");
}
}

public static void functionRunner(int functionChecker,int rowIndex,String

```

```

row_letter,int conditionController){
    /*
     * Runs the specific function depending on the option selected
     * @param Row letter in uppercase,option selected by the user, row
index, row letter entered by the user, condition controller
    */
    if(functionChecker==1)
        cancelProgress(row_letter, rowIndex,conditionController);
    else if (functionChecker==2)
        buyingProgress(row_letter,conditionController,rowIndex);
    else if (functionChecker==3)
        specificTicketFinder(row_letter,rowIndex,conditionController);

}
public static void switcher(int functionChecker){
    /*
     * Gets the row letter from the user and calls the functionRunner
method and
     * sending relevant arguments according to each row.
     * @params integer controlling which method to be called
    */
    try {
        System.out.print("Please Enter the row letter: ");
        String row_letter=new Scanner(System.in).next().toLowerCase();
        switch (row_letter) {
            case "a":
                functionRunner(functionChecker,0,row_letter,15);
                break;
            case "b":
                functionRunner(functionChecker,1,row_letter,13);
                break;
            case "c":
                functionRunner(functionChecker,2,row_letter,13);
                break;
            case "d":
                functionRunner(functionChecker,3,row_letter,15);
                break;
            default:
                System.out.println(row_letter.toUpperCase()+" row doesn't
Exist!");
        }
    } catch (Exception e) {
        System.out.println("Some Error Occurred: "+e);
    }
}
public static void buyingProgress(String row_letter,int controller,int
rowIndex){
    /*
     * Gets seat number and allows the user to buy the seat
     * @params ,row letter entered by the user , condition controller
    */
    System.out.print("Enter the Seat number: ");
    try {
        int seat_number = new Scanner(System.in).nextInt();
        if(seat_number>0 && seat_number<controller){
            if(seats[rowIndex][seat_number-1]==0){
                boolean state=false;

```

```

        String email=null;
        System.out.print("\nPlease Enter your name: ");
        String name = new Scanner(System.in).next();
        System.out.print("Please Enter your surname: ");
        String surname = new Scanner(System.in).next();
        while(!state){
            System.out.print("Please Enter your email: ");
            email = new Scanner(System.in).next();
            if(email.contains("@") && email.contains(".")){
                state=true;
            }else{
                System.out.println("Invalid email please enter a
valid email.");
            }
        }
        seats[rowIndex][seat_number-1]=1;
        double price;
        if(seat_number<6) price=200;
        else if(seat_number<10) price=150;
        else price=180;
        Person person = new Person(name,surname,email);
        Ticket ticket = new Ticket(row_letter.toUpperCase(),
        seat_number-1, price, person);
        if (ticketCount < 52) {
            tickets[ticketCount] = ticket;
            ticketCount++;
            ticket.save();
            System.out.println("Seat "+row_letter.toUpperCase()+""
"+seat_number+" Ticket bought successfully!");
        }
        }else{
            System.out.println("Seat "+row_letter.toUpperCase()+""
"+seat_number+" is not available.");
        }
        }else{
            System.out.println("Enter a correct Seat number.");
        }
    } catch (InputMismatchException e) {
        System.out.println("Enter an Integer");
    } catch (Exception e){
        System.out.println("Error Occurred\nError code: "+e);
    }
}
public static void cancelProgress(String row_letter,int row_index,int
conditionController) {
/*
    Gets seat number and cancels the ticket and deletes the file
    @params row letter, row index and the controller for the conditions
*/
try {
    System.out.print("Enter the Seat number: ");
    int seat_number = new Scanner(System.in).nextInt();
    if (seat_number > 0 && seat_number < conditionController) {
        if (seats[row_index][seat_number - 1] == 1) {
            seats[row_index][seat_number - 1] = 0;
            for (int i = 0; i < ticketCount; i++) {

```

```

        Ticket ticket = tickets[i];
        if (Objects.equals(ticket.getRow(),
row_letter.toUpperCase()) && ticket.getSeat() + 1 == seat_number) {
            System.out.println("Ticket cancelled
Successfully");
            ticket.delete();
            for (int j = i; j < ticketCount - 1; j++) {
                tickets[j] = tickets[j + 1];
            }
            tickets[ticketCount - 1] = null;
            ticketCount--;
        }
    }
} else {
    System.out.println("Seat " + row_letter.toUpperCase() + "
" + seat_number + " is available already.");
}
else {
    System.out.println("Enter a correct Seat number.");
}
} catch (InputMismatchException e) {
    System.out.println("Incorrect input, Enter an Integer input! ");
} catch (Exception e) {
    System.out.println("Some error occurred \nError code:" + e);
}
}
public static void main(String[] args) {
/*
     Main method initializing the program and calling the relevant
methods
*/
System.out.print("\n\tWelcome to the Plane Management System");
seats[0] = new int[14];
seats[1] = new int[12];
seats[2] = new int[12];
seats[3] = new int[14];
int response;
do{
    response=user_menu();
    switch(response){
        case 0: System.out.println("Thank you for using Plane
Management.");break;
        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;
        default: System.out.println("Invalid Option,Try
again..");break;
    }
}while (response!=0);
}
}

```

<<Ticket.java >>

```
// Ticket.java
import java.io.File;
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) {
        /*
         * Assigns the values received as @params to the instance variables
         * @params row, seat number, price, person object
        */
        this.row = row;
        this.seat = seat;
        this.price = price;
        this.person = person;
    }

    public String getRow() {
        return row;
    }

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

    public int getSeat() {
        return seat;
    }

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

    public double getPrice() {
        return price;
    }

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

    public Person getPerson() {
        return person;
    }
}
```

```

}

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

public void save() {
    /*
        Saves the ticket information into a text file
    */
    String filename = row + (seat+1) + ".txt"; // Generate filename based
on row and seat number
    try (FileWriter writer = new FileWriter(filename)) {
        writer.write("Person Information:\n");
        writer.write("Name: " + person.getName() + "\n"+ "Surname: " +
person.getSurName() + "\n" + "Email: " + person.getEmail() + "\n");
        writer.write("Ticket Information:\n" + "Row: " + row + "\n" + "Seat:
" + (seat + 1) + "\n" + "Price: " + price + "\n");
        System.out.println("Ticket information saved to file: " +
filename);
    } catch (IOException e) {
        System.out.println("Error occurred while saving ticket
information to file: " + e.getMessage());
    }
}
public void delete(){
    /*
        deletes the text file containing the user information
    */

    try {
        File file=new File(row+(seat+1)+".txt");
        if (!file.delete()) {
            System.out.println("Failed to delete the ticket information
stored inside the file!");
        }
    } catch (Exception e) {
        throw new RuntimeException(e);
    }
}

public void printInfo() {
    /*
        Prints the user info
    */
    System.out.println("Person Information:");
    person.printInfo();
    System.out.print(" Row: " + row);
    System.out.println(" Seat: " + (seat+1));
    System.out.println(" Price: " +price);
}
}

```

<<Person.java >>

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

    public Person(String name, String surname, String email) {
        /*
            Assigns the values received as @params to the instance variables
            @params name,surname & email
        */
        this.name = name;
        this.surname = surname;
        this.email = email;
    }

    // Getters and setters
    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    // Method to print Person information
    public void printInfo() {
        /*
            Prints person's information
        */
        System.out.println("  Name: " + name);
        System.out.println("  Surname: " + surname);
    }
}
```

```
        System.out.println(" Email: " + email);  
    }  
}
```

<<END>>