Informatics Institute of Technology

School of Computing

Software Development II Coursework Report

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

Date of submission : 3/23/2024

Student ID : < 20231564 > / < w2052169 >

Student First Name : Chamith

Student Surname : Wijewantha

Tutorial group (day, time, and tutor/s): G-15 / Tuesday / 10.30-12.30 / Mr.Torin Weerasinghe&

"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 : Chamith Sandeepa Wijewantha

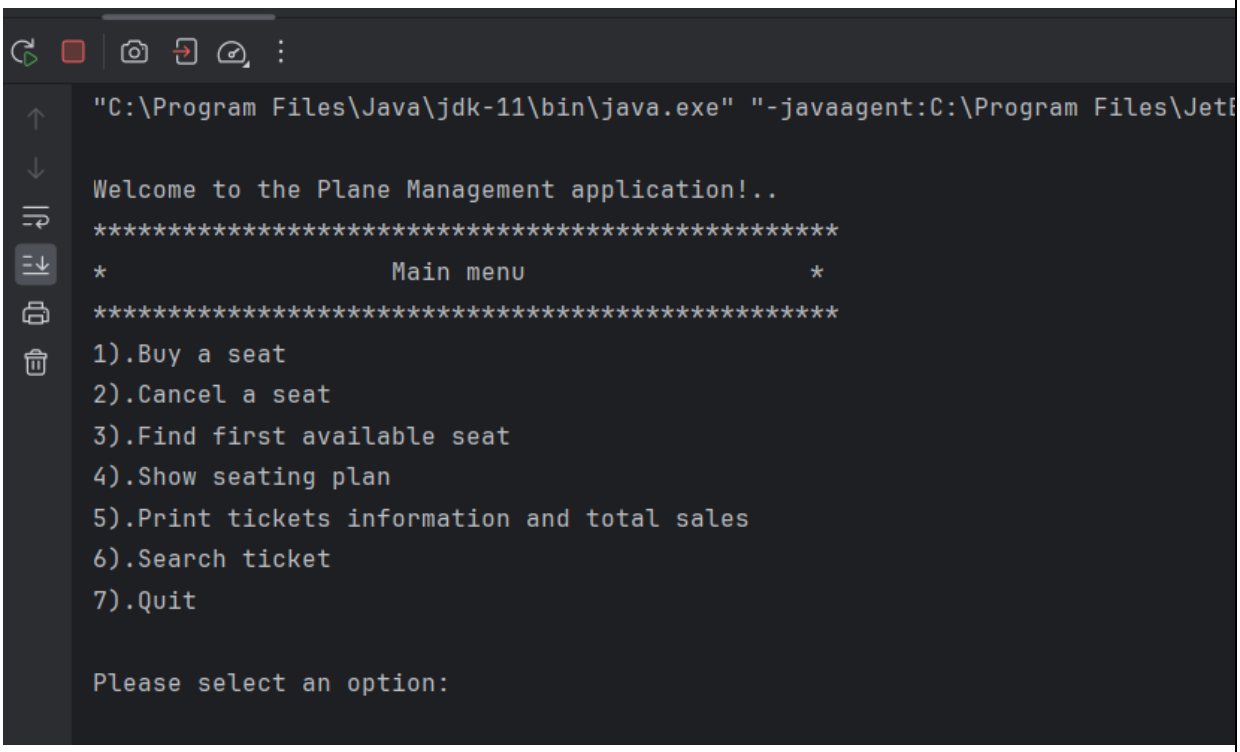
Student ID : 20231564

Self-assessment form and test plan

1) Self-assessment form

Task	Self-assessment (select one)	Comments
1	☒Fully implemented ☐Partially implemented ☐Not attempted	Task 1 is done and fully implemented.
2	☒Fully implemented ☐Partially implemented ☐Not attempted	In my program, use the string array and give all options. Then use an enhanced for loop and display the user menu beginning of the program.

Insert here a screenshot of your welcome message and menu:



3	☒Fully implemented ☐Partially implemented ☐Not attempted	In task 3, get the two inputs, such as row letter and seat number from the user and check whether conditions are true or not,if conditions are true seat book is successful, and, if conditions are false system displays “Invalid seat!..Seat is not found!....”.
4	☒Fully implemented ☐Partially implemented ☐Not attempted	In this method doing,when we are booked a seat previously,in here can cancel this seat,this process are doing here.
5	☒Fully implemented ☐Partially implemented ☐Not attempted	We can inwalk this method by entering the number 3 as an option. We can find what is the first available seat in this plane according to the seat plan. It is very useful for the user to see what is the first available seat in a plane without seeing show seating plane.
6	☒Fully implemented ☐Partially implemented ☐Not attempted	In this method, the user can see what is the seating plan in this plane at the very beginning of the program or, until the program user can see what are the available seats and not available seats in this plane. Apart from that user can verify whether his seat is booked successfully or not!...
Insert here a screenshot of the seating plan:		

```
Please select an option:
4
==Now you can see seating plan==

Seating 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
```

7	☒Fully implemented ☐Partially implemented ☐Not attempted	In this Person class, through the getters and setters store the data when user enters.
8	☒Fully implemented ☐Partially implemented ☐Not attempted	This method, prints all the information of the person and ticket prints through data access by getters.
9	☒Fully implemented ☐Partially implemented ☐Not attempted	Previous buy_seat method extends and gets the name, surname, and email as input from a person. The previous cancel_seat method extends and when the user cancels a ticket, it removes the ticket from the array list of tickets.
10	☒Fully implemented ☐Partially implemented ☐Not attempted	This method prints the information of all tickets that have been sold during the session, additionally, it prints the total number of sold tickets and

		the total amount of tickets were sold during the session.
11	☒Fully implemented ☐Partially implemented ☐Not attempted	In this method user can input a row letter and seat number and search if someone has bought that seat or not yet. Apart from that users can see details of booked seats.
12	☒Fully implemented ☐Partially implemented ☐Not attempted	Ticket that saves the information of the ticket in a file. File Is saves name of the row and the seat number. Eg:- (When the user booked, row 'A' seat number 1, the file will save as A1.txt)

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**
When run the programme (Task 2)	-	Display menu and, prompt for the user input.	Welcome to the Plane Management application!.. ********** * Main menu * ********** 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 7).Quit Please select an option:	☒Pass ☐Fail
Task 3). User enter the number 1 for his option	1	Display "==Now you can buy a seat==" and prompt for the row letter: seat number:	==Now you can buy a seat== Please enter the row letter: Please enter the seat number:	☒Pass ☐Fail

Enter invalid row letter and seat number as input.	E 2	Want to show an error message like this: “Invalid seat numbers. Please check again.” And display the menu options again.	“Invalid seat number!.. Please check again.” 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 7).Quit	☒Pass ☐Fail
Enter valid row letter and seat number as input. You enter this valid row letter and seat number again as a input.	A 1 A 1	Display a message like this: “Seat booked successfully!” Display a message like this: “Sorry!. Already seat is not available.”	Display a message like this: “Seat booked successfully!” Display a message like this: “Sorry!. Already seat is not available.”	☒Pass ☐Fail
Task 4). The user enter the number 2 for his option	2	Display “==Now you can cancel a seat==” And prompt for the enter row letter and seat number:	==Now you can cancel a seat== Please enter the row letter: Please enter the seat number:	☒Pass ☐Fail
Enter invalid row letter and seat number as input.	Q 1	Display a error message like this: “Invalid seat!.. Seat is not found!....”	“Invalid seat!..Seat is not found!....”	☒Pass ☐Fail

Enter valid row letter and seat number as input when you bought in previous step.	A 1	Display a message like this: “Seat has been cancelled successfully!”	“Seat has been cancelled successfully!”	☒Pass ☐Fail
Task 5). User enter the number 3 for his option	3	Display a message like this: “==Now you can find first available seat==”	“==Now you can find first available seat==” “First available seat: Row A, Seat 1”	☒Pass ☐Fail
Task 6). User enter the number 4 for his option in beginning	4	Display a message “Seating Plan” and display seating plan.	“Seating Plan” O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O	☒Pass ☐Fail
The user go to the buy seat option and successfully buys seat then enter the number 4 his option, then user can see his seat is successfully booked and it include the	First, go to the buy seat option and buy row A first seat!.. Then enter number 4 as a input	Display a message “Seating Plan” and display a seating plan with the included booked sheet.	Seating Plan: X O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O	☒Pass ☐Fail

seating plan successfully.				

==========

Part B testing

Test case / scenario	Input	Expected Output	Output	Pass/Fail
Task 7) Create a new class file called Person and add a method that prints the information from person.	___	Print the information about the person with ticket information when the user enters the number ‘5’ as an option.	Customer's name: Customer's surname: Customer's email:	☒Pass ☐Fail
Task 8). Create a new class file called Ticket with row, seat number and Person.	___	Print the information about the person with ticket information when the user enters the number ‘5’ as an option.	Row number: Seat number: Seat price: Person information: Customer's name: Customer's surname: Customer's email:	☒Pass ☐Fail

Task 9). 1). Extend the buy seat method, when buying a ticket, it asks for all information of a Person. Insert your first name: Insert your surname: Insert your email:	If you are entered the valid correct input as a input like this: Insert your first name: Chamith Insert your surname: Sandeepa Insert your email: c123@gmail.com	Display a message like this: “All information enterd successfully!..” & “Seat booked successfully!”	Display a message like this: “All information enterd successfully!..” & “Seat booked successfully!”	☒Pass ☐Fail
In the above test case, you entered an invalid incorrect name, the system shows the error message and says to reinsert the name again.	/ . ‘ @ 4	Display a message like this: “**Invalid name!**” “Insert name again:”	Display a message like this: “**Invalid name!**” “Insert name again:”	☒Pass ☐Fail
you entered an invalid incorrect surname, the system	/ . ‘	Display a message like this: “**Invalid surname!**” “Insert surname again:”	Display a message like this: “**Invalid surname!**” “Insert surname again:”	☒Pass ☐Fail

shows the error message and says to reinsert the surname again.	@ 4			
you entered an invalid email, the system shows the error message and says to reinsert the email again.	C2662gmail.com c123,,,@gmail.com q123@gmailcom cqw122../3@gmail.com	Display a message like this: "**Invalid email. Check and insert again!..**" "Inset email again:"	Display a message like this: "**Invalid email. Check and insert again!..**" "Inset email again:"	☒Pass ☐Fail
Task 9) 2). Extend the cancel seat method such that when cancelling a seat a ticket removes the ticket from the array list of ticket.	in the beginning you buy Row A seat number 1, now you give these Row letter and seat number as the output in here… Row letter: A Seat number: 1	Display a message like this: "Seat has been cancelled successfully!"	Display a message like this: "Seat has been cancelled successfully!"	☒Pass ☐Fail
We want to ensure seat is cancelled successfully, we can enter this row letter	(in search ticket method) Row letter: A Seat number: 1	In search ticket method shows the message like this: "This seat is available."	n search ticket method shows the message like this: "This seat is available."	☒Pass ☐Fail

and seat number in search ticket option!....				
Task 10). print_tickets_info that prints the information of all tickets that have been sold during the session.	In the beginning, you buy Row A seat number 1, and gives these inputs in there Insert your first name: Chamith Insert your surname: Sandeepa Insert your email: c@gmail.com then you enter number '5' in here.. input : '5'	Row number:A Seat number:1 Seat price:200.0 Person information: Customer's name:chamith Customer's surname:sandeepa Customer's email:cs@gmail.com ========== Total tickets that have been sold during the session:1 Total sales: £200.0 ========== ==========	Row number:A Seat number:1 Seat price:200.0 Person information: Customer's name:chamith Customer's surname:sandeepa Customer's email:cs@gmail.com ========== Total tickets that have been sold during the session:1 Total sales: £200.0 ========== ==========	☒Pass ☐Fail
Task 11). Method search ticket asks the user to input a row letter and a seat number and searches if someone has bought that seat.	In the beginning, you buy Row A seat number 1, and give these inputs in there Row letter: A Seat number: 1	Display ticket information like this: ==Ticket found!== Row number:A Seat number:1 Seat price:200.0 Person information: Customer's name:chamith Customer's surname:sandeepa Customer's email:cs@gmail.com ========== ====	==Ticket found!== Row number:A Seat number:1 Seat price:200.0 Person information: Customer's name:chamith Customer's surname:sandeepa Customer's email:cs@gmail.com ========== ====	☒Pass ☐Fail

Task 12). Add a method **Save** in the class ticket that saves the information of the ticket every time a ticket is sold.	In the beginning, you buy Row A seat number 1, then create a file the name of the row and the seat number.	Display a message like this ,when end of the buy seat, “Ticket information are successfully saved to file: A1.txt” And, A1.txt file actually create in the file location which when we given.	Display a message like this ,when end of the buy seat, “Ticket information are successfully saved to file: A1.txt” And, Including all customer details and A1.txt file successfully created on the file location “w2052169_PlaneManagement” . In file:- ========== TICKET INFORMATION: ========== Row: A Seat: 1 Price: 200.0 ========== Person Information: ========== Name: chamith Surname: sandeepa Email: cs@gmail.com	☒Pass ☐Fail

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

3) **Code :**

Part A:-

```
import java.io.File;

import java.util.InputMismatchException;

import java.util.Scanner;



public class Main {

    //Database area.Using 1D arrays for the store data row by row.

    static String[] row_A = new String[14];

    static String[] row_B = new String[12];

    static String[] row_C = new String[12];

    static String[] row_D = new String[14];



    //declare Ticket array for store all information related to the Ticket class.

    static Ticket[] tickets = new Ticket[52];

    static int ticketsSold = 0;  //set the ticketSold count as Zero in beginning.

    //these public attributes can use not only in class but also another classes.
```

```
    public static String name;

    public static  String surname;

    public static String email;

    public static double price;

    public static Person person;




    public static void main(String[] args) {

        System.out.println();

        System.out.println("Welcome to the Plane Management application!..");

        System.out.println("*".repeat(50));

        System.out.println("* "+" ".repeat(18)+"Main menu"+" ".repeat(18)+" *");

        System.out.println("*".repeat(50));

        Scanner input = new Scanner(System.in);


        //Display the following menu and ask the user to select an option until user enter number '7'
as a option.

        boolean loop = true;

        while(loop){

            String [] startMenu={            //using string array
```

```
            "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",
            "7).Quit\n"
        };


    //using enhanced for loop and display the string array with starting menu.
        for (String menu : startMenu) {
            System.out.println(menu);
        }
        //this try catch valid for the all the methods which content the try block...
        //User enter the invalid input as a input then not system crash, it catch by catch block and returns a error message.
        try {
            System.out.println("Please select an option (1-7):");
            int option = input.nextInt();
            //using switch case and inwalk the methods.
```

```
switch (option) {
    case 1:
        System.out.println("==Now you can buy a seat==\n");
        buy_seat();
        break;
    case 2:
        System.out.println("==Now you can cancel a seat==\n");
        cancel_seat();
        break;
    case 3:
        System.out.println("==Now you can find first available seat==\n");
        find_first_available();
        break;
    case 4:
        System.out.println("==Now you can see seating plan==\n");
        show_seating_plan();
        break;
    case 5:
        System.out.println("==Now you can see tickets information and total sales==\n");
        print_tickets_info();
```

```
                break;
            case 6:
                System.out.println("==Now you can search ticket==\n");
                show_seating_plan();
                search_ticket();
                break;
            case 7:
                System.out.println("You are exiting the Plane Management application!\n Good Bye!.....");
                loop = false;
                break;
            default:
                System.out.println("Invalid input!..Please check the Main Menu again!\n");
                break;
        }
    }catch (InputMismatchException ex){
        System.out.println("Error occured!..Please enter the valid input.");
        input.nextLine();
    }
}
```

```
}
public static boolean buy_seat() {
    Scanner input = new Scanner(System.in);
    System.out.println("Please enter the row letter (A-D):");
    //Getting row letter as a input and store it as a char value.
    char rowLetter = input.next().toUpperCase().charAt(0);



    System.out.println("Please enter the seat number:");
    // getting seat number as a input from user and store it as a int value.
    int seatNumber = input.nextInt();


    boolean validSeatnumber = false;
    switch(rowLetter){
        case 'A':
            validSeatnumber = seatNumber>=1 && seatNumber<=row_A.length;
            break;
        case 'B':
            validSeatnumber = seatNumber>=1 && seatNumber<=row_B.length;
            break;
```

```
    case 'C':
        validSeatnumber = seatNumber>=1 && seatNumber<=row_C.length;
        break;
    case 'D':
        validSeatnumber = seatNumber>=1 && seatNumber<=row_D.length;
        break;
    default:
        validSeatnumber = false;
}
if(!validSeatnumber){
    System.out.println("Invalid seat number!..Please check again.");
    return false;
}


boolean seatAvailable = false;
switch (rowLetter){
    case 'A':
        seatAvailable = row_A[seatNumber-1]==null;
        break;
    case 'B':
```

```
        seatAvailable = row_B[seatNumber-1]==null;
        break;
    case 'C':
        seatAvailable = row_C[seatNumber-1]==null;
        break;
    case 'D':
        seatAvailable = row_D[seatNumber-1]==null;
        break;
    default:
        seatAvailable = false;
}
if(!seatAvailable){
    System.out.println("Sorry!..Already seat is not available.");
    return false;
}
System.out.println("Insert customer details:\n");


boolean isValid =false;
System.out.println("Insert your first name:");
//using while loop and loops it again and again until when user insert valid correct inputs.
```

```
while(!isValid){
    name = input.next();
    //using java Regular Expressions and check weather user insert valid input or not...
    //[a-zA-Z] it can check name content is only alphabetical letters..
    if(!name.matches("[a-zA-Z]+")){
        System.out.println("**Invalid name!..**");
        System.out.println("Insert name again: ");
    }else{
        isValid = true;
    }
}


isValid = false;
System.out.println("Insert your surname:");
while(!isValid){
    surname = input.next();
    if(!surname.matches("[a-zA-Z]+")){
        System.out.println("**Invalid surname!..**");
        System.out.println("Insert surname again: ");
    }else{
```

```
            isValid = true;
        }
    }


    isValid =false;
    System.out.println("Insert your email:");
    while(!isValid){
        email = input.next();
        //using java Regular Expressions and check weather user insert valid email or not...
        if(!email.matches("^[a-zA-Z0-9_+&*-]+(?:\\.[a-zA-Z0-9_+&*-]+)*@(?:[a-zA-Z0-9-]+\\.)+[a-zA-Z]{2,7}$")){
            System.out.println("**Invalid email..Check and insert again!..**");
            System.out.println("Inset email again:");
        }else {
            isValid = true;
        }
    }


    System.out.println("All information enterd successfully!..\n");
    //name, surname, and email values are used to initialize the properties of the Person object being created.
```

```
//person is the reference variable of Person class

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



switch (rowLetter){

    case 'A':

        row_A[seatNumber - 1] = "booked";

        break;

    case 'B':

        row_B[seatNumber - 1] = "booked";

        break;

    case 'C':

        row_C[seatNumber - 1] = "booked";

        break;

    case 'D':

        row_D[seatNumber - 1] = "booked";

        break;

}



// Create and store the ticket

price = ticketPrice(seatNumber);
```

```
    Ticket ticket = new Ticket(rowLetter, seatNumber, price, person);

    //all data set and store the ticket array

    tickets[ticketsSold] = ticket;

    //sold tickets counter

    ticketsSold++;



    System.out.println("Seat booked successfully!");

    ticket.save();

    return true;

}

//set the ticket price using seat number

public static double ticketPrice(int seatNumber){

    double price = 0;

    if(seatNumber>=1 && seatNumber<=5){

        price = 200;

    }else if(seatNumber>=6 && seatNumber<=9){

        price = 150;

    }else{

        price = 180;

    }
```

```
        return price;
    }
    public static boolean cancel_seat(){
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter the row letter:");
        char rowLetter = input.next().toUpperCase().charAt(0);
        System.out.println("Please enter the seat number:");
        int seatNumber = input.nextInt();

        String filename = rowLetter+String.valueOf(seatNumber)+".txt";

        File deleteFile = new File(filename);

        Ticket ticket = new Ticket(rowLetter, seatNumber, price, person);

        switch (rowLetter) {
            case 'A':
                if (seatNumber < 1 || seatNumber >= row_A.length) { //check seat number is inrange or
out of range
                    System.out.println("Inavalid seat number!...Check seeting plan again!..\n");
```

```
        } else if (row_A[seatNumber - 1] == null) { //check relevant seat number is null(available)
            System.out.println("There's no need to cancel,the seat is available yet!..\n");
        } else if (row_A[seatNumber - 1] != null) {  //check relevant seat number is !null(not available)
            row_A[seatNumber - 1] = null;
            if (deleteFile.exists()) {
                deleteFile.delete();  //delete file from file location.
                System.out.println("Seat has been cancelled successfully!\n");
            }
        }
        for (int i = 0; i <ticketsSold; i++) {
            if(tickets[i]!=null){  //loop the ticket array and check what are the !null places to check conditions.
                if(tickets[i].getRow() == rowLetter && tickets[i].getSeat() == seatNumber){
                    tickets[i]=null;  //when cancel the seat,remove the all data from tickets array and clear it.
                    break;
                }
            }
        }
```

```
    break;
case 'B':
    if (seatNumber < 1 || seatNumber >= row_B.length) {
        System.out.println("Inavalid seat number!...Check seeting plan again!..\n");
    } else if (row_B[seatNumber - 1] == null) {
        System.out.println("There's no need to cancel,the seat is available yet!..\n");
    } else if (row_B[seatNumber - 1] != null) {
        row_B[seatNumber - 1] = null;
        if (deleteFile.exists()) {
            deleteFile.delete();
            System.out.println("Seat has been canceled successfully!\n");
        }
    }
    for (int i = 0; i <ticketsSold; i++) {
        if(tickets[i]!=null){
            if(tickets[i].getRow() == rowLetter && tickets[i].getSeat() == seatNumber){
                tickets[i]=null;
                break;
            }
        }
```

```
    }
    break;
case 'C':
    if (seatNumber < 1 || seatNumber >= row_C.length) {
        System.out.println("Inavalid seat number!...Check seeting plan again!..\n");
    } else if (row_C[seatNumber - 1] == null) {
        System.out.println("There's no need to cancel,the seat is available yet!..\n");
    } else if (row_C[seatNumber - 1] != null) {
        row_C[seatNumber - 1] = null;
        if (deleteFile.exists()) {
            deleteFile.delete();
            System.out.println("Seat has been canceled successfully!\n");
        }
    }
    for (int i = 0; i <ticketsSold; i++) {
        if(tickets[i]!=null){
            if(tickets[i].getRow() == rowLetter && tickets[i].getSeat() == seatNumber){
                tickets[i]=null;
                break;
            }
```

```
            }
        }
        break;
    case 'D':
        if (seatNumber < 1 || seatNumber >= row_D.length) {
            System.out.println("Inavalid seat number!...Check seeting plan again!..\n");
        } else if (row_D[seatNumber - 1] == null) {
            System.out.println("There's no need to cancel,the seat is available yet!..\n");
        } else if (row_D[seatNumber - 1] != null) {
            row_D[seatNumber - 1] = null;
            if (deleteFile.exists()) {
                deleteFile.delete();
                System.out.println("Seat has been canceled successfully!\n");
            }
        }
        for (int i = 0; i <ticketsSold; i++) {
            if(tickets[i]!=null){
                if(tickets[i].getRow() == rowLetter && tickets[i].getSeat() == seatNumber){
                    tickets[i]=null;
                    break;
```

```
                }
            }
        }
        break;
    default:
        System.out.println("Invalid seat!..Seat is not found!....\n");
    }
    return true;
}
public static boolean find_first_available(){
    //find the first available seat using by for loops
    for (int i = 0; i <=row_A.length; i++) { //using for i loop and loop row_A array and find what is the first null place in array.
        if (row_A[i] == null) {
            System.out.println("First available seat: Row A, Seat " + (i+1)+"\n");
            return true;
        }
    }
    for (int i = 0; i <=row_B.length; i++) {
        if(row_B[i] == null){
```

```
            System.out.println("First available seat: Row B,Seat " + (i+1)+"\n");
            return true;
        }
    }
    for (int i = 0; i <=row_C.length; i++) {
        if(row_C[i] == null){
            System.out.println("First available seat: Row C,Seat " + (i+1)+"\n");
            return true;
        }
    }
    for (int i = 0; i <=row_D.length; i++) {
        if(row_D[i] == null){
            System.out.println("First available seat: Row D,Seat " + (i+1)+"\n");
            return true;
        }
    }
    return true;
}
public static boolean show_seating_plan(){
    System.out.println("Seating Plan:\n");
```

```
        for (int i = 0; i <row_A.length; i++) {
            System.out.print(row_A[i] != null ? " X " : " O "); //check weather null or not, if it is !null then prints 'X' and if it is=null, then prints 'O'.
        }
        System.out.println(); //This line prints output to the console without starting a new line.


        for (int i = 0; i <row_B.length; i++) {
            System.out.print(row_B[i]!= null? " X " : " O ");
        }
        System.out.println();
        for (int i = 0; i <row_C.length; i++) {
            System.out.print(row_C[i]!= null? " X " : " O ");
        }
        System.out.println();
        for (int i = 0; i <row_D.length; i++) {
            System.out.print(row_D[i]!= null? " X " : " O ");
        }
        System.out.println("\n");
        return true;
    }
```

```
public static boolean print_tickets_info(){
    double totalamountofTicket=0;
    if(ticketsSold == 0){
        System.out.println("No any ticket that have been sold during the session!..\n");
        return false;
    }
    //using for loop and loop tickets array and check below conditions,when these conditions are true then prints the ticket information.
    for (int i = 0; i < tickets.length; i++) {
        Ticket ticket = tickets[i];
        if (ticket != null) {
            ticket.printTicketinformation();
            //calculate total amount of tickets which sold tickets.
            totalamountofTicket += ticket.getPrice();
            System.out.println("============================");
        }
    }
    System.out.println("Total tickets that have been sold during the session:"+ticketsSold);
    System.out.println("Total sales: £" + totalamountofTicket);
    System.out.println("============================");
```

```
        return true;
    }
    public static boolean search_ticket(){
        Scanner input = new Scanner(System.in);
        System.out.println("Now you are in search ticket option.");
        System.out.println("Please enter the row letter(A-D):");
        char rowLetter = input.next().toUpperCase().charAt(0);
        System.out.println("Please enter the seat number:");
        int seatNumber = input.nextInt();
        System.out.println();


        boolean found = false;
        for (Ticket ticket : tickets) { //using enhanced for loop and loop the ticket array
            //check until null situation and then check row letter and seat number are equals,then it equals prints the ticket information.
            if (ticket != null  && ticket.getRow() == rowLetter && ticket.getSeat() == seatNumber) {
                found = true;
                System.out.println("==Ticket found!==");
                ticket.printTicketinformation();
```

```
            System.out.println("=======================");
            break;
        }
    }
    //conditions when not true, it says the seat is not yet buy anyone and its available already..and then prints("This seat is available.")
    if (!found) {
        System.out.println("This seat is available.\n");
    }
    return true;
  }

}
```

Part B:-

```
public class Person {
  //class attributes
  private String name;
  private  String surname;
```

```
private  String email;


//parameterized constructor
public Person(String name, String surname, String 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;
}

//create printInformation method to print customers data
public void printInformation(){
    System.out.println("Customer's name:"+ name);
    System.out.println("Customer's surname:"+ surname);
    System.out.println("Customer's email:"+ email);
}
```

```
}

import java.io.File;
import java.io.FileWriter;
import java.io.IOException;

public class Ticket {

    //class attributes
    private char row;
    private int seat;
    private double price;
    private Person person;

    //default constructor
    public Ticket() {

    }

    //parameterized constructor
```

```
public Ticket(char row, int seat, double price, Person person) {
    this.row = row;
    this.seat = seat;
    this.price = price;
    this.person = person;
}

//getters and setters
public char getRow() {
    return row;
}

public void setRow(char 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;
```

```
    }


    //in this save method create a file and prints all customers details and ticket information in one file.
    public void save(){
        //set the file name
        String filename = row+String.valueOf(seat)+".txt";
        //File obj = new File(filename);
        try{
            FileWriter myWriter = new FileWriter(row+String.valueOf(seat)+".txt");
            myWriter.write("=======================\n");
            myWriter.write("TICKET INFORMATION:\n");
            myWriter.write("=======================\n");
            myWriter.write("Row: "+row+"\n");
            myWriter.write("Seat: " + seat + "\n");
            myWriter.write("Price: " + price + "\n");
            myWriter.write("=======================\n");
            myWriter.write("Person Information:\n");
            myWriter.write("=======================\n");
            myWriter.write("Name: " + person.getName() + "\n");
```

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

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

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

            myWriter.close();

            System.out.println("Ticket information are successfully saved to file: "+(filename));

System.out.println("=====================================================
===");




        }catch (IOException e){

            System.out.println("An error occured!.....");



        }



    }

    public void printTicketinformation(){

        System.out.println("Row number:"+ row);

        System.out.println("Seat number:"+ seat);

        System.out.println("Seat price:"+ price);

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

```
        person.printInformation();

    }

}
```

<<END>>