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 : 24/03/2024

Student ID : 20230282 / w2051616

Student First Name : Nethini

Student Surname : Perera

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

Group : 11

Lecture : Mr. Saman Hettiarachchi (Thursday : 10.30 – 12.30)

Tutorial – Ms. Rashmi Perera, Mr. Shiyam Baasith (Wednesday : 8.30 – 10.30)

"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 : S.A. Nethini Pabodhya Perera

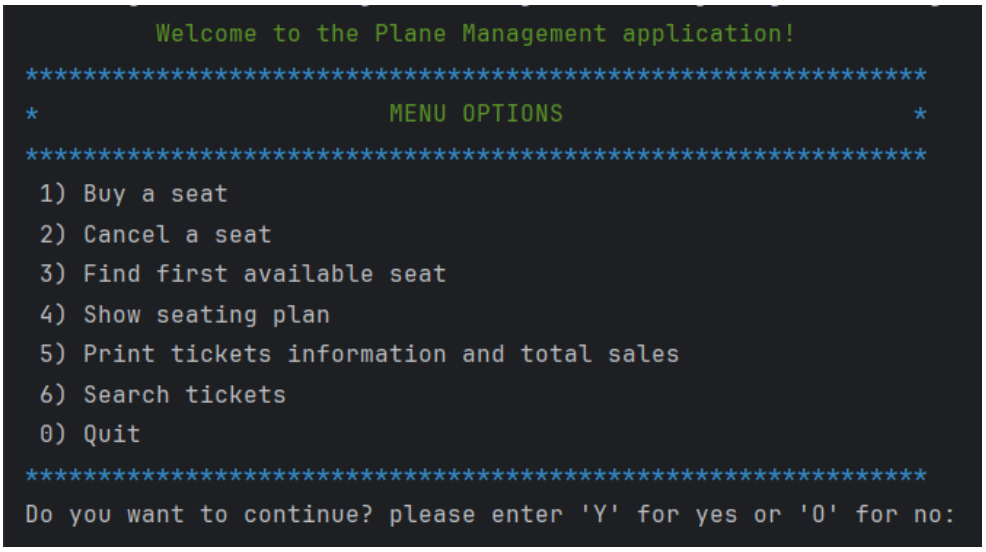
Student ID : 20230282

Self-assessment form and test plan

1) Self-assessment form

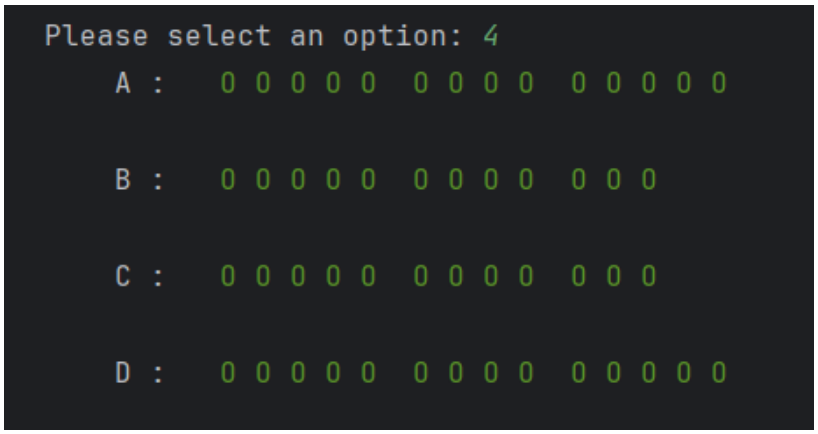
Task	Self-assessment (select one)	Comments
1	☒Fully implemented ☐Partially implemented ☐Not attempted	Created a new project titled as w2051616_PlaneManagement. Then, create a class file named w2051616_PlaneManagement.java. This class contain a main method. The seat management system was implemented by using standard arrays.
2	☒Fully implemented ☐Partially implemented ☐Not attempted	A user menu was added to the main method, which asks the user to select an option. Entering character '0' would exit the program without crashing or giving an error in the program.

Insert here a screenshot of your welcome message and menu:



3	☒Fully implemented ☐Partially implemented ☐Not attempted	Create a method called buy_seat. When the user enters option 1 in the main menu, the program will ask for the row letter and the seat number. if that seat is available, it will print it; if it is not, it will print that seat is already booked.
4	☒Fully implemented ☐Partially implemented ☐Not attempted	Create a method called cancel_seat. When the user enters option 2 in the main menu, the program will ask for the row letter and the seat number. Then checked that the seat was already booked or not. If the seat is booked, then it will cancel the seat. if it is not, display it as the seat is available.
5	☒Fully implemented ☐Partially implemented ☐Not attempted	Create a method called find_first_available. When the user selects '3' in the main menu, it will show the first available seat. That means it will search the seat in order to seat row A, B,C and then row D.
6	☒Fully implemented ☐Partially implemented ☐Not attempted	Create a method called show_seating_plan. when the user selects '4' in the main menu, it shows the seats that are available and the seats that have been sold.

Insert here a screenshot of the seating plan:



7	☒Fully implemented ☐Partially implemented ☐Not attempted	Create a new class file named as Person.java. This contains the attributes named name, surname, and email. Also, all the getters and the setters were added to this class.
8	☒Fully implemented ☐Partially implemented ☐Not attempted	Create a new class file named Ticket.java. this contains the attributes which are named as seat row, seat number, ticket price and person information.
9	☒Fully implemented ☐Partially implemented ☐Not attempted	Create a new array of tickets. Buy_seat method was extended and here asks the user to input person information. Also, cancel_seat method was extended and here removes that seat from the array.
10	☒Fully implemented ☐Partially implemented ☐Not attempted	Create a method called print_ticket_information. when the user selects '5' in the main menu, this will print all the information about each ticket that has been booked and the total sales of all of those tickets.
11	☒Fully implemented ☐Partially implemented ☐Not attempted	Create a method called search_ticket. when the user selects '6' in the main menu, this asks the user to enter a row letter and a seat number. If the seat is already booked, this will print the ticket buyer's personal information. If it is not, it will print that the seat is available.
12	☒Fully implemented ☐Partially implemented ☐Not attempted	Create a method called write_info, which saves the information on the ticket (including the person's information) in a file. This method is called every time a ticket is sold.

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
Enter “Y” in the starting message.	Do you want to continue? please enter 'Y' for yes or 'O' for no: Y	Please select an option:	Please select an option:	☒Pass ☐Fail
Enter “O” in the starting message.	Do you want to continue? please enter 'Y' for yes or 'O' for no: O	Please select an option:	Please select an option:	☒Pass ☐Fail
Enter any other character other than “O” or “Y”.	Do you want to continue? please enter 'Y' for yes or 'O' for no: mm8	Invalid please enter 'Y' for yes or 'O' for no:	Invalid please enter 'Y' for yes or 'O' for no:	☒Pass ☐Fail
Enter “1” as an option menu.	Please select an option: 1	please enter a row letter as your seat row:	please enter a row letter as your seat row:	☒Pass ☐Fail
Enter a character in the main menu.	Please select an option: g	Invalid Option Do you want to start again? please enter 'Y' for yes or 'O' for no: y	Invalid Option Do you want to start again? please enter 'Y' for yes or 'O' for no: y	☒Pass ☐Fail
Enter a number which is out of range in the main menu.	Please select an option: 123	Invalid Option Do you want to start again? please enter 'Y' for yes or 'O' for no: y	Invalid Option Do you want to start again? please enter 'Y' for yes or 'O' for no: y	☒Pass ☐Fail
Enter a valid row letter.	please enter a row letter as your seat row: A	Please enter a seat number:	Please enter a seat number:	☒Pass ☐Fail

Enter an integer or any other character other than A,B,C,D.	please enter a row letter as your seat row: s or please enter a row letter as your seat row: 12	Invalid Row Letter please enter a row letter as your seat row:	Invalid Row Letter please enter a row letter as your seat row:	☒Pass ☐Fail
Enter a seat number which is available.	Please enter a seat number: 5	Seat is available. Please enter your name :	Seat is available. Please enter your name :	☒Pass ☐Fail
Enter a seat which is already booked.	Please enter a seat number: 5	Seat was already booked. Please select an option:	Seat was already booked. Please select an option:	☒Pass ☐Fail
Enter a seat number which is out of range.	Please enter a seat number: 45	seat number is out of range Please enter a seat number:	seat number is out of range Please enter a seat number:	☒Pass ☐Fail
Enter a character as the seat number.	Please enter a seat number: g	Integer Required Please enter a seat number:	Integer Required Please enter a seat number:	☒Pass ☐Fail
Enter “2” as an option menu.	please enter a row letter as your seat row: A	Please enter a seat number:	Please enter a seat number:	☒Pass ☐Fail
Enter an already booked seat when calling the 2nd option in the main menu.	please enter a row letter as your seat row: A Please enter a seat number: 5	successfully deleted the ticket information. The seat has been successfully cancelled.	successfully deleted the ticket information. The seat has been successfully cancelled.	☒Pass ☐Fail
Enter an already available seat when calling the 2nd option in the main menu.	please enter a row letter as your seat row: B Please enter a seat number: 8	seat is already not available. Please select an option:	seat is already not available. Please select an option:	☒Pass ☐Fail

Firstly buy A1, A2, A3, B6, C1, A9 seats and then enter “3” in the main menu.	Please select an option: 3	The first available seat in Seat Row A The seat number is : 4 Please select an option:	The first available seat in Seat Row A The seat number is : 4 Please select an option:	☒Pass ☐Fail
Enter “4” in the main menu.	Please select an option: 4	A : X X X O O O O O O O O O B : O O O O O O O O O O O C : O O O O O O O O O O O D : O O O O O O O O O O O O Please select an option:	A : X X X O O O O O O O O O B : O O O O O O O O O O O C : O O O O O O O O O O O D : O O O O O O O O O O O O Please select an option:	☒Pass ☐Fail
Enter “O” in the main menu.	Please select an option: O	END Process finished with exit code 0	END Process finished with exit code 0	☒Pass ☐Fail

Part B testing

Test case / scenario	**Input**	**Expected Output**	**Output**	**Pass/Fail**
Enter “5” as an option menu as the very first choice.	Please select an option: 5	Total Sales of tickets : £0.0 Please select an option:	Total Sales of tickets : £0.0 Please select an option:	☒Pass ☐Fail

Firstly, buy seat A8 and then enter “5” into the main menu.	Please select an option: 1 please enter a row letter as your seat row: A Please enter a seat number: 8 Please select an option: 5	Name : nethini Surname : perera Email : nethini720@gmail.com Seat Row : A Seat number : 8 Ticket Price : 150.0 Total Sales of tickets : £150.0	Name : nethini Surname : perera Email : nethini720@gmail.com Seat Row : A Seat number : 8 Ticket Price : 150.0 Total Sales of tickets : £150.0	☒Pass ☐Fail
Firstly, buy seat A8 and then buy D14 seat and after that enter “5” in the main menu.	Please select an option: 1 please enter a row letter as your seat row: A Please enter a seat number: 8 Please select an option: 5 Please select an option: 1 please enter a row letter as your seat row: D Please enter a seat number: 14	Please select an option: 5 Name : nethini Surname : perera Email : nethini720@gmail.com Seat Row : A Seat number : 8 Ticket Price : 150.0 Name : shenali Surname : fernando Email : sheni12@gmail.com Seat Row : D Seat number : 14 Ticket Price : 180.0 Total Sales of tickets : £330.0	Please select an option: 5 Name : nethini Surname : perera Email : nethini720@gmail.com Seat Row : A Seat number : 8 Ticket Price : 150.0 Name : shenali Surname : fernando Email : sheni12@gmail.com Seat Row : D Seat number : 14 Ticket Price : 180.0 Total Sales of tickets : £330.0	☒Pass ☐Fail
Enter “6” in the main menu , then input a row letter and seat number that is not booked already.	Please select an option: 6 please enter a row letter as your seat row: B Please enter a seat number: 4	This seat is available Please select an option:	This seat is available Please select an option:	☒Pass ☐Fail

Enter “6” in the main menu , then input a row letter and seat number that is already booked .	Please select an option: 1 please enter a row letter as your seat row: A Please enter a seat number : 8	Name : nethini Surname : perera Email : nethini720@gmail.com Seat Row : A Seat number : 8 Ticket Price : 150.0	Name : nethini Surname : perera Email : nethini720@gmail.com Seat Row : A Seat number : 8 Ticket Price : 150.0	☒Pass ☐Fail
Once a person books a ticket, then all the ticket and personal information are saved to a file which is created before.		successfully saved	successfully saved	☒Pass ☐Fail
Once a person cancels a ticket, then all the ticket and personal information which is saved before a file is deleted.		successfully deleted the ticket information	successfully deleted the ticket information	☒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 :

w2051616_PlaneManagement.java

/*"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 : S. A. Nethini Pabodhya Perera

 Student ID : 20230282 / w2051616 */



import java.util.InputMismatchException;

import java.util.Scanner;

//Define a class.

public class w2051616_PlaneManagement {

    // Create a Scanner object for user inputs.

    public static Scanner input = new Scanner(System.in);


    // Create an arrays to represent seat rows A, B, C, and D.

    public static int[][] seatrow_A = {{0, 0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0, 0}};

    public static int[][] seatrow_B = {{0, 0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0}};

    public static int[][] seatrow_C = {{0, 0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0}};

    public static int[][] seatrow_D = {{0, 0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0, 0}};

    // Create an array to store Ticket objects.
```

```
public static Ticket[] ticket_array = new Ticket[52];


// A method to display seating rows.
private static void seatrow_array(int[][] seatrow, String row) {
    System.out.print("    " + row + ": ");
    for (int p = 0; p < seatrow.length; p++) {
        System.out.print(" ");
        for (int r = 0; r < seatrow[p].length; r++) {
            if ((seatrow[p][r]) == 0) {      // A condition to display available seats.
                System.out.print( Green+ " O" + Reset);
            } else if ((seatrow[p][r]) == 1) {     // A condition to display booked seats.
                System.out.print(   Blue + " X" + Reset);
            }


        }
    }
    System.out.println(" \n ");
}
```

```
// A method to display the entire seating plan.
private static void show_seating_plan() {
    //
    seatrow_array(seatrow_A, " A ");
    seatrow_array(seatrow_B, " B ");
    seatrow_array(seatrow_C, " C ");
    seatrow_array(seatrow_D, " D ");
}

// Variable declarations.
public static String row_letter;
public static int seat_number;
public static double ticket_price;

// A method to select a seat row.
private static void select_seatrow() {
    System.out.print(" please enter a row letter as your seat row: ");
```

```
    row_letter = input.next();
    switch (row_letter) {
        case "A":
            break;
        case "B":
            break;
        case "C":
            break;
        case "D":
            break;
        default:
            System.out.println("Invalid Row Letter");
            select_seatrow();     // Recursive call if an invalid row letter is entered.
    }
}

// A method to select a seat number.
private static void select_seatnumber() {
```

```
while (true) {    // A while loop for recalling the same method.
    try {
        System.out.print(" Please enter a seat number: ");
        seat_number = input.nextInt();
        if ((row_letter.equals("A")) || (row_letter.equals("D"))) {
            if ((seat_number > 0) && (seat_number <= 14)) {      // A condition to check the correct range.
                break;
            } else {
                System.out.println("seat number is out of range");
                continue;
            }

        } else if ((row_letter.equals("B")) || (row_letter.equals("C"))) {
            if ((seat_number > 0) && (seat_number <= 12)) {
                break;
            } else {
                System.out.println("seat number is out of range");
```

```
                continue;
            }
        }
    } catch (InputMismatchException e) {        //exceptional handling for validations.
        System.out.println("Integer Required");
        input.nextLine();
    } catch (Exception e) {
        input.nextLine();
    }
  }
}

// Main method.
public static void main(String[] args) {

    // Display welcome message and menu options.
    System.out.println( Green + "          Welcome to the Plane Management application!     " + Reset);
```

```
        System.out.println(                                Blue                                +
"************************************************************"    +
Reset);

        System.out.println( Blue + "*" + Reset + Green +"                              MENU
OPTIONS                    "+ Reset + Blue +"*" + Reset);

        System.out.println(                                Blue                                +
"************************************************************"    +
Reset);

        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 tickets ");

        System.out.println(" 0) Quit ");

        System.out.println(                                Blue                                +
"************************************************************"    +
Reset);

        System.out.print("Do you want to continue? please enter 'Y' for yes or 'O' for
no: ");

        String decision = input.next();
```

```
boolean Continue = true;
while (Continue) {
    try {
        if (decision.equalsIgnoreCase("Y")) {
            System.out.print(" Please select an option: ");    // Prompt user to select an option
            String option = input.next();
            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 ("O") :                    // Quit the program
                System.out.println(Blue + "END" + Reset);
                Continue = false;
                break;
            default:
                System.out.print( Green + " Invalid Option" + Reset +" \n Do you want to start again? please enter 'Y' for yes or 'O' for no: ");
                decision = input.next();
                break;
        }
```

```
            } else if (decision.equalsIgnoreCase("O")) {
                System.out.println(Blue + "END" + Reset);
                Continue = false;
                break;
            } else {
                System.out.print(" Invalid \n  please enter 'Y' for yes or 'O' for no:");
                decision = input.next();
            }
        } catch (InputMismatchException e) {
            System.out.println(Green + "Integer Required" + Reset);
            input.nextLine();
        } catch (Exception e) {
            input.nextLine();
        }
    }
}
```

```
// A method to prompt user to add details when adding a ticket
private static void add_ticket() {
    input.nextLine();
    System.out.print("Please enter your name : ");
    String Name = input.nextLine();
    System.out.print("Please enter your surname : ");
    String Surname = input.nextLine();
    System.out.print("Please enter your Email-Address  : ");
    String Email = input.nextLine();

    // Create a Person object with person information.
    Person personInfo = new Person();
    personInfo.setName(Name);
    personInfo.setSurname(Surname);
    personInfo.setEmail(Email);

    // Create a Ticket object with seat and person information
    Ticket ticketInfo = new Ticket();
```

```
        ticketInfo.setRow(row_letter);
        ticketInfo.setSeat_number(seat_number);
        ticketInfo.setPerson_info(personInfo);
        ticketInfo.setTicket_price(ticket_price);

        // A loop to find an empty seat to add the ticket in ticket array
        for (int i = 0; i < ticket_array.length; i++) {
            if (ticket_array[i] == null) {
                ticket_array[i] = ticketInfo;   // Add the ticket to the first available seat in the ticket array.
                ticketInfo.write_info();   // Save ticket and person information to a file.
                break;
            }
        }
    }

    // Method to display tickets information and total sales.
    private static void print_tickets_info() {
```

```
        double Total_Sales = 0;
        for (Ticket element : ticket_array) {    // Iterate through the ticket array
            if (element != null) {
                element.display_ticket_info();   // Display information of each ticket
                Total_Sales += element.getTicket_price();    // Add each ticket price to total sales
            }
        }
        System.out.println(Blue + "Total Sales of tickets : " + "£" + Total_Sales + Reset); // Display total sales.
    }


    private static void buy_seat() {     // Method to buy a seat
        select_seatrow();     //calling select seat row method.
        select_seatnumber();   //calling select seat number method.

        // Check which row the seat belongs to user inputs and calling if seat available method.
```

```
        if (row_letter.equals("A")) {
            if_seat_available(seatrow_A);
        } else if (row_letter.equals("B")) {
            if_seat_available(seatrow_B);
        } else if (row_letter.equals("C")) {
            if_seat_available(seatrow_C);
        } else if (row_letter.equals("D")) {
            if_seat_available(seatrow_D);
        }
    }

    // Method to check if the selected seat is available
    private static void if_seat_available(int[][] array) {
        if ((seat_number > 0) && (seat_number <= 5)) {
            if ((array[0][seat_number - 1]) == 0) {       // A condition to display that the seat is available.
                System.out.println(Green + " Seat is available" + Reset);
                (array[0][seat_number - 1]) = 1;    // Marked that available seat booked.
```

```
            ticket_price = 200;    // Assigning a value to ticket price.
            add_ticket();   //calling the method add ticket.
        } else {
            System.out.println(Blue + "Seat was already booked" + Reset);   // Display that the seat is already booked.
        }
    } else if ((seat_number > 5) && (seat_number <= 9)) {
        if ((array[1][seat_number - 6]) == 0) {
            System.out.println(Green + " Seat is available" + Reset);
            (array[1][seat_number - 6]) = 1;
            ticket_price = 150;
            add_ticket();
        } else {
            System.out.println(Blue + "Seat was already booked" + Reset);
        }
    } else if ((seat_number > 9) && (seat_number <= 14)) {
        if ((array[2][seat_number - 10]) == 0) {
            System.out.println(Green + " Seat is available" + Reset);
```

```
            (array[2][seat_number - 10]) = 1;
            ticket_price = 180;
            add_ticket();
        } else {
            System.out.println(Blue + "Seat was already booked" + Reset);
        }
    } else {
        System.out.println(Green + "seat number is out of range" + Reset);
    }
}


private static void cancel_seat() {    // Method to cancel a seat
    select_seatrow();     //calling select seat row method.
    select_seatnumber();   //calling select seat number method.

    // Check which row the seat belongs to user inputs and calling if seat already booked method.
```

```
        if (row_letter.equals("A")) {
            if_seat_alreadyBooked(seatrow_A);
        } else if (row_letter.equals("B")) {
            if_seat_alreadyBooked(seatrow_B);
        } else if (row_letter.equals("C")) {
            if_seat_alreadyBooked(seatrow_C);
        } else if (row_letter.equals("D")) {
            if_seat_alreadyBooked(seatrow_D);
        }
    }

    // Method to cancel a ticket and person information.
    private static void cancel_ticket() {
        for (int i = 0; i < ticket_array.length; i++) {
            if (ticket_array[i] != null && ticket_array[i].getRow().equals(row_letter) && (ticket_array[i].getSeat_number()) == seat_number) {
                ticket_array[i].deleteInfo_file();      // Delete ticket information that saved before to a file.
                ticket_array[i] = null;          // Remove ticket from array
```

```
            break;
        }
    }
}

// Method to check if the selected seat is already booked
private static void if_seat_alreadyBooked(int[][] Array) {
    if ((seat_number > 0) && (seat_number <= 5)) {
        if ((Array[0][seat_number - 1]) == 1) {     // A condition to display that the seat is already booked.
            (Array[0][seat_number - 1]) = 0;     //Marked that booked seat is available.
            cancel_ticket();         //calling the method cancel ticket.
            System.out.println(Blue + " The seat has been successfully cancelled " + Reset);
        } else {
            System.out.println(Blue + "seat is already not available" + Reset);       // Display seat is already not available

        }
```

```
    } else if ((seat_number > 5) && (seat_number <= 9)) {
        if ((Array[1][seat_number - 6]) == 1) {
            (Array[1][seat_number - 6]) = 0;
            cancel_ticket();
            System.out.println(Blue + " The seat has been successfully cancelled " +
Reset);
        } else {
            System.out.println(Blue + "seat is already not available" + Reset);
        }
    } else if ((seat_number > 9) && (seat_number <= 14)) {
        if ((Array[2][seat_number - 10]) == 1) {
            (Array[2][seat_number - 10]) = 0;
            cancel_ticket();
            System.out.println(Blue + " The seat has been successfully cancelled " +
Reset);
        } else {
            System.out.println(Blue + "seat is already not available" + Reset);
        }
    } else {
```

```
        System.out.println(Green+ "seat number is out of range" + Reset);
    }


}


public static int k;   //variable declaration.
public static int m;


// Method to find the first available seat in a given seat row.
private static void find_the_array(int[][] newarray) {


    // Loop through each row and column in the seat row.
    outerLoop:
    for (k = 0; k < 3; k++) {
        innerLoop:
        // Check if the seat is available (marked as 0).
        for (m = 0; m < newarray[k].length; m++) {
            if (newarray[k][m] == 0) {
```

```
                statement = "available";          // Set the statement to indicate the availability.
                break outerLoop;          // Break out of the outer loop once an available seat is found.
            } else {
                statement = "not_available";          // If seat is not available, set the statement.
            }
        }
    }
}

// Method to display the found available seat.
private static void found() {
    if (k == 0) {
        System.out.println(Blue + " The seat number is : " + (m + 1) + Reset);    // Display seat which starts numbering from 1
    } else if (k == 1) {
        System.out.println(Blue + " The first available seat number is : " + (m + 6) + Reset);  // Display seat which starts numbering from 6.
```

```
        } else if (k == 2) {
            System.out.println(Blue + " The first available seat number is : " + (m + 10)
+ Reset);  // Display seat which starts numbering from 10.
        }
    }

    public static String statement;   // variable declaration.
    public static String x;

    // Method to find the first available seat in any seat row.
    private static void find_first_available() {
        x = "True";       // Control variable
        if (x.equals("True")) {
            find_the_array(seatrow_A);    // Check Seat Row A available that seat.
            if (statement.equals("available")) {
                System.out.println(Blue + " The first available seat in Seat Row A" +
Reset);
                found();    // Calling found method.
            } else if (statement.equals("not_available")) {
```

```
            find_the_array(seatrow_B);     // Check Seat Row B available that seat.
            if (statement.equals("available")) {
               System.out.println(Blue + " The first available seat in Seat Row B" +
Reset);
               found();     // Calling found method.
            } else if (statement.equals("not_available")) {
               find_the_array(seatrow_C);     // Check Seat Row C available that seat.
               if (statement.equals("available")) {
                  System.out.println(Blue + " The first available seat in Seat Row C" +
Reset);
                  found();     // Calling found method.
               } else if (statement.equals("not_available")) {
                  find_the_array(seatrow_D);      // Check Seat Row D available that
seat.
                  if (statement.equals("available")) {
                     System.out.println(Blue + " The first available seat in Seat Row
D" + Reset);
                     found();     // Calling found method.
                  } else {    // Display that all the seats are booked.
```

```
                    System.out.println(Green + " SORRY \n All the seats are already booked in this plane" + Reset);
                }
            }
        }
    }
}

// Method to search for a ticket.
private static void search_ticket() {
    select_seatrow();     //calling select seat row method.
    select_seatnumber();     //calling select seat number method.
    if (row_letter.equals("A")) {
        search_if_booked(seatrow_A);      // Search if the seat is in Seat Row A is booked.
    } else if (row_letter.equals("B")) {
        search_if_booked(seatrow_B);       // Search if the seat in Seat Row B is booked.
```

```
        } else if (row_letter.equals("C")) {
            search_if_booked(seatrow_C);        // Search if the seat in Seat Row C is booked.
        } else if (row_letter.equals("D")) {
            search_if_booked(seatrow_D);        // Search if the seat in Seat Row D is booked.
        }
    }

    // Method to check if a seat is booked and display information.
    private static void search_if_booked(int[][] searcharray) {
        if ((seat_number > 0) && (seat_number <= 5)) {
            if ((searcharray[0][seat_number - 1]) == 1) {
                for (int i = 0; i < ticket_array.length; i++) {
                    if (ticket_array[i] != null && ticket_array[i].getRow().equals(row_letter) && (ticket_array[i].getSeat_number()) == seat_number) {
                        ticket_array[i].display_ticket_info();      // Display ticket information if the seat is booked.
                        break;
```

```
            }
        }

    } else if ((searcharray[0][seat_number - 1]) == 0) {
        System.out.println( Blue +  "This seat is available" + Reset);       // If seat is not booked, indicate availability
    }
} else if ((seat_number > 5) && (seat_number <= 9)) {
    if ((searcharray[1][seat_number - 6]) == 1) {
        for (int i = 0; i < ticket_array.length; i++) {
            if (ticket_array[i] != null && ticket_array[i].getRow().equals(row_letter) && (ticket_array[i].getSeat_number()) == seat_number) {
                ticket_array[i].display_ticket_info();
                break;
            }
        }

    } else if ((searcharray[1][seat_number - 6]) == 0) {
```

```
            System.out.println( Blue +  "This seat is available" + Reset);
        }
    } else if ((seat_number > 9) && (seat_number <= 14)) {
        if ((searcharray[2][seat_number - 10]) == 1) {
            for (int i = 0; i < ticket_array.length; i++) {
                if (ticket_array[i] != null && ticket_array[i].getRow().equals(row_letter) && (ticket_array[i].getSeat_number()) == seat_number) {
                    ticket_array[i].display_ticket_info();
                    break;
                }
            }

        } else if ((searcharray[2][seat_number - 10]) == 0) {
            System.out.println( Blue +  "This seat is available" + Reset);
        }
    }
}
public static String Green = "\u001B[32m";
```

```
    public static String Blue = "\u001B[34m";
    public static String Reset = "\u001B[0m";
}
```

Person.java

```
// A Java class which is named as "Person".
public class Person {
    private String name;  // Here are private instance variables which represent the name, surname, and email of the person.
    private String surname;
    private String email;



    // Here is a default constructor method with no parameters.
    public Person (){
        this.name = "Not Defined";
        this.surname = "Not Defined";
        this.email = "Not Defined";
    }
```

```
// Here is a constructor method with parameters for name, surname, and email.
public Person(String name, String surname, String email){
    this.name = name;
    this.surname = surname;
    this.email = email;
}
// These are getter methods.
// A method which returns the name of the person.
public String getName(){
    return this.name;
}

// A method which returns the surname of the person.
public String getSurname(){
    return this.surname;
}
```

```
// A method which returns the email of the person.
public String getEmail(){
    return this.email;
}

// These are setter methods.
// A method which sets the name of the person.
public void setName(String name){
    this.name = name;
}

// A method which sets the surname of the person.
public void setSurname(String surname){
    this.surname = surname;
}

// A method which sets the email of the person.
public void setEmail(String email){
```

```
        this.email = email;
    }

    // A method which prints the information of the person.
    public void Person_information(){
        System.out.println("Name : " + getName());
        System.out.println("Surname : " + getSurname());
        System.out.println("Email : " + getEmail() );
    }
}
```

Ticket.java

```
import java.io.File;
import java.io.IOException;
import java.io.FileWriter;
// A Java class which is named as Ticket
public class Ticket {
```

private String Row; // Here are private instance variables which represent the seat row, seat number, ticket price and person information.

```
    private int Seat_number;
    private double Ticket_price;
    private Person person_info;

    // Here is a default constructor method with no parameters.
    public Ticket() {
        this.Row = "Not defined";
        this.Seat_number = 0;
        this.Ticket_price = 0;
        this.person_info = null;
    }

    // Here is a constructor method with parameters for seat row, seat number, ticket price and person information.
    public Ticket(String Row, int Seat_number, double Ticket_price, Person person_info) {
        this.Row = Row;
        this.Seat_number = Seat_number;
        this.Ticket_price = Ticket_price;
```

```
    this.person_info = person_info;
}

// These are getter methods.
// A method which returns the name of the ticket row.
public String getRow() {
    return this.Row;
}

// A method which returns the name of the ticket seat number.
public int getSeat_number() {
    return this.Seat_number;
}

// A method which returns the name of the ticket price.
public double getTicket_price() {
    return this.Ticket_price;
}
```

```
    // Here is a setter method for retrieving the person information associated with the ticket.
    public Person getPerson_info() {
        return this.person_info;
    }

    // Here is a setter method for setting the seat row.
    public void setRow(String Row) {
        this.Row = Row;
    }

    // Here is a setter method for setting the seat number.
    public void setSeat_number(int Seat_number) {
        this.Seat_number = Seat_number;
    }

    // Here is a setter method for setting the ticket price.
```

```
public void setTicket_price(double Ticket_price) {
    this.Ticket_price = Ticket_price;
}

// Here is a setter method for setting the person information.
public void setPerson_info(Person person_info) {
    this.person_info = person_info;
}

// A method to display information about the ticket and the person.
public void display_ticket_info() {
    getPerson_info().Person_information();
    System.out.println("Seat Row : " + getRow());
    System.out.println("Seat number : " + getSeat_number());
    System.out.println("Ticket Price : " + getTicket_price());
    System.out.println();
}
```

```
    // A method to write ticket information to a text file when ticket is booked.
    public void write_info() {
        try {
            FileWriter file_save = new FileWriter("C:\\Users\\ASUS\\Desktop\\Viva\\"+ getRow() + getSeat_number() + ".txt");
            file_save.write("Seat row :  " + this.getRow() + "\n");
            file_save.write("Seat number : " + this.getSeat_number() + "\n");
            file_save.write("Ticket price : " + this.getTicket_price() + "\n");
            file_save.write("Name : " + this.getPerson_info().getName() + " " + this.getPerson_info().getSurname() + "\n");
            file_save.write("Email Address : " + getPerson_info().getEmail() + "\n");
            file_save.close();
            System.out.println("successfully saved");
        } catch (IOException e) {
            System.out.println("Error occurred " + e.getMessage());
        }

    }
```

```
    // A method to delete the ticket information file if ticket canceled.
    public void deleteInfo_file() {
        File folder = new File("C:\\Users\\ASUS\\Desktop\\Viva\\"+ getRow() + getSeat_number() + ".txt");
       try {
          if (folder.exists()) {
             folder.delete();
             System.out.println("successfully deleted the ticket information");
          } else {
             System.out.println("Folder does not exist");
          }
       } catch (Exception e) {
          System.out.println("Error occurred ");
       }

    }
}
```

<<END>>