UOW-logo

Informatics Institute of Technology

School of Computing

Software Development II Coursework Report

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

Date of submission : 25/03/2024

Student ID : 20231160 / w2051972

Student First Name : Anjana

Student Surname : Ranasinghe

Tutorial group (day, time, and tutor/s): Group - 14

"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 : A. Anjana Senath Ranasinghe

Student ID : 20231160 / w2051972

## Self-assessment form and test plan

# Self-assessment form

|  |  |  |
| --- | --- | --- |
| Task | Self-assessment (select one) | Comments |
| 1 | Fully implemented  Partially implemented  Not attempted | Project created with the given title and given class file names. Implemented the seat management system using standard arrays. |
| 2 | Fully implemented  Partially implemented  Not attempted | Added the user menu to the programme and call the methods. |
| Insert here a screenshot of your welcome message and menu: | | |
| 3 | Fully implemented  Partially implemented  Not attempted | Implemented the method called “buy\_seat” that asks the user to input a row letter and a seat Number. Checked that the row and seat entered are valid and that the seat is available. User can buy it and updated in the arrays. |
| 4 | Fully implemented  Partially implemented  Not attempted | Created a method called “cancel\_seat” that cancel a seat. It should ask the user to input a row number and a seat number. Then cancelled the seat otherwise if it is available prompt user that “Seat is already Available”. |
| 5 | Fully implemented  Partially implemented  Not attempted | That find the first seat which is still available. Then the programme display the relevant row letter and the seat number. |
| 6 | Fully implemented  Partially implemented  Not attempted | Implemented the method called “show\_seating\_plan” that shows the seats that are available and the seats that have been sold. Display available seats with the character ‘O’ and the sold seats with ‘X’. |
| Insert here a screenshot of the seating plan: | | |
| 7 | Fully implemented  Partially implemented  Not attempted | Created the “Person” class with the  name, surname, and email attributes. Added a constructor that takes the 3 variables as input to create an object Person. Added all the getters of the class Person. |
| 8 | Fully implemented  Partially implemented  Not attempted | Created a new class file called “Ticket” with the row, seat, price, and Person attributes to store ticket information. |
| 9 | Fully implemented  Partially implemented  Not attempted | Added a object array named tickets to store the sold tickets. Extended the “cancel\_seat” method such that when cancelling a ticket, it removes, the ticket from the array list of tickets. |
| 10 | Fully implemented  Partially implemented  Not attempted | That prints the information of all  tickets that have been sold. And calculates the total price of the tickets sold during the session. |
| 11 | Fully implemented  Partially implemented  Not attempted | Created a method called “search\_ticket” that asks the user to input a row letter and a seat number and searches if someone has bought that seat. If someone has bought the seat, it will print the Ticket and Person information. Otherwise, will display “This seat is Available!”. |
| 12 | Fully implemented  Partially implemented  Not attempted | Added a method “save” in the class Ticket that saves the information of the Ticket including the Person’s information in a text file. |

# 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 a seat | Menu Option - 1  Row letter - A  Seat num – 1 | Sold the A1 seat | Expected Output | Pass  Fail |
| Cancel a seat | Menu Option – 2  Row letter – A  Seat num - 1 | Cancelling the sold seat | Expected Output | Pass  Fail |
| Find the first available seat | Menu Option - 3 | Prompting the first available seat A1. | Expected Output | Pass  Fail |
| **Show seating**  plan | Menu Option - 4 | Showing the seat plan according to the plan. | Expected Output | Pass  Fail |
| buy a seat | Menu Option - 1  Row letter – A  Seat num – 20 | Display the user to incorrect seat number. | Expected Output | Pass  Fail |
| buy a seat | Menu Option - 1  Row letter – E | Display the user to incorrect seat Row. | Expected Output | Pass  Fail |
| Execute the programme | Menu Option - 0 | Exiting the  Programme. | Expected Output | Pass  Fail |

## Part B testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case / scenario | Input | Expected Output | Output | Pass/Fail |
| buy a seat | Menu Option - 1  Row letter - A  Seat num – 10  Firstname – Anjana  Surname –  Ranasinghe  Email –  [anjana@gmail.com](mailto:anjana@gmail.com) | Save the person  information to the “Person” object and then to the “Ticket”  object array. | Expected Output | Pass  Fail |
| **Print ticket**  information | Menu option - 5 | Display all the information relevant to the reserved seats. | Expected Output | Pass  Fail |
| Search ticket | After buying the seat A10  Menu option - 6  Seat row - A  Seat number - 10 | Display sold information relevant to the seat A10. | Expected Output | Pass  Fail |
| Search ticket | After buying the seat A10  Menu option - 6  Seat row - A  Seat number - 5 | Display seat is available. | Expected Output | Pass  Fail |
| buy a seat | Menu Option - 1  Row letter - A  Seat num – 10  Firstname – Anjana  Surname –  Ranasinghe  Email –  [anjanagmail.com](mailto:anjana@gmail.com) | Display Incorrect email address format. | Expected Output | Pass  Fail |
| Search ticket | After buying the seat A10  Menu option - 6  Seat row - A  Seat number – g | Display seat number is invalid and ask user to again enter the  correct value. | Expected Output | Pass  Fail |
| Save Files | Menu option - 1  Seat row - A  Seat number - 5  Name - Anjana  Surname –Senath  Email –  [anjana@gmail.com](mailto:anjana@gmail.com) | Already saved the file by “A5.txt” with ticket and person’s information. | Expected Output | 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.**

1. **Code :**

**PlanManagement.java**

**import java.util.Scanner; //import the Scanner class to read inputs**

**public class PlanManagement {**

**private static int[][] array = new int[4][]; //Array to store seat availability**

**private static Ticket[] tickets = new Ticket[52]; //Array to store sold tickets**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**//initialize arrays for seat availability**

**array[0] = new int[14];**

**array[1] = new int[12];**

**array[2] = new int[12];**

**array[3] = new int[14];**

**boolean True = true;**

**while (True) {**

**try{**

**int option;**

**do{**

**displayMenu(); //To display the main menu**

**System.out.print("Please select an Option :");**

**option = scanner.nextInt();**

**True = false;**

**//Switch statement to handle user options**

**switch (option) {**

**case 1 -> buy\_seat(scanner); //To buy a seat**

**case 2 -> cancel\_seat(scanner); //To cancel a seat**

**case 3 -> find\_first\_available(); //Find the first available seat**

**case 4 -> show\_seating\_plan(); //Shows the seats that are available(O) and the seats that have been sold(X)**

**case 5 -> print\_tickets\_info(); //Prints the information of all tickets that have been sold**

**case 6 -> search\_ticket(scanner); //Search the ticket from person's information**

**}**

**System.out.println();**

**}while(option !=0);**

**System.out.println("\*\*\*\* THANK YOU ! \*\*\*\*");**

**} catch(Exception ex){**

**System.out.println("Invalid choice. Please try again. ");**

**scanner.next(); //Clear the buffer**

**True = true;**

**}**

**}**

**}**

**//Method to display the main menu**

**private static void displayMenu(){**

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

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

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

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

**System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");**

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

**System.out.println("\* MENU OPTIONS \*");**

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

**System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");**

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

**System.out.println("1) buy a seat");**

**System.out.println("2) Cancel a seat");**

**System.out.println("3) Find first available seat");**

**System.out.println("4) Show seating plan");**

**System.out.println("5) Print tickets information and total sales");**

**System.out.println("6) Search ticket");**

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

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

**System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");**

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

**}**

**//Method to buy a seat**

**public static void buy\_seat(Scanner scanner){**

**//asks the user to input a row letter**

**System.out.print("Enter row letter (A,B,C or D): ");**

**char rowletter = scanner.next().toUpperCase().charAt(0);**

**int rownum = (int) (rowletter - 'A');**

**//check the row letter is correct or not**

**if (rownum < 0 || rownum >= array.length) {**

**System.out.println("Invalid row letter!");**

**buy\_seat(scanner); //Call to allow the user to input the row letter again**

**return;**

**}**

**int seatnum = 0;**

**boolean correct = true;**

**while (correct) {**

**try{**

**//asks the user to input a seat number**

**System.out.print("Enter seat number : ");**

**seatnum = scanner.nextInt();**

**correct = false;**

**//validation**

**if (seatnum<0 || seatnum > array[rownum].length) {**

**System.out.println("Invaild seat number!");**

**correct = true;**

**}**

**} catch(Exception x){**

**System.out.println("Invaild seat number!");**

**scanner.next(); //Clear the buffer**

**}**

**}**

**if (array[rownum][seatnum-1] == 1) {**

**System.out.println("\nSeat is already Sold.");**

**} else{**

**//Prompt user to confirm seat purchase**

**System.out.print("\nSeat is available! Do you want to buy it? (yes or no): ");**

**String confirm = scanner.next().toLowerCase();**

**if (confirm.equals("yes")) {**

**array[rownum][seatnum-1] = 1;**

**// asks the user to input personal details**

**System.out.print("Enter your first name: ");**

**String firstname = scanner.next();**

**System.out.print("Enter your sure name: ");**

**String surname = scanner.next();**

**String email = null;**

**boolean True = true;**

**while (True) {**

**System.out.print("Enter your valid email address: ");**

**email = scanner.next();**

**//check the email address format**

**if (email.contains("@") && email.contains(".")) {**

**True = false;**

**} else{**

**System.out.println("Incorrect email address format!(e.g- example@example.com) ");**

**}**

**}**

**//Create a Person object with user input**

**Person person = new Person(firstname, surname, email);**

**//check the ticket price based on seat number**

**double price = 0;**

**if (seatnum >= 1 && seatnum <=5 ) {**

**price = 200;**

**}**

**else if (seatnum >= 6 && seatnum <=9 ) {**

**price = 150;**

**}**

**else{**

**price = 180;**

**}**

**//adding the ticket array to find available slot and creat ticket object**

**for(int i = 0; i< tickets.length; i++){**

**if (tickets[i] == null) {**

**tickets[i] = new Ticket(rowletter, seatnum, price, person);**

**tickets[i].save(); //Save ticket information to file**

**break;**

**}**

**}**

**System.out.println("\nYour seat reservation Successfully Sold. Safe Flight! ");**

**} else{**

**System.out.println("\nSeat not Sold. ");**

**}**

**}**

**}**

**//Method to cancel a seat**

**private static void cancel\_seat(Scanner scanner){**

**//asks the user to input a row letter**

**System.out.print("Enter row letter (A,B,C or D): ");**

**char rowletter = scanner.next().toUpperCase().charAt(0);**

**int rownum = (int) (rowletter - 'A');**

**//validation**

**if (rownum < 0 || rownum >= array.length) {**

**System.out.println("Invalid row letter!");**

**cancel\_seat(scanner); //Call to allow the user to input the row letter again**

**return;**

**}**

**int seatnum = 0;**

**boolean correct = true;**

**while (correct) {**

**try{**

**//asks the user to input a seat number**

**System.out.print("Enter seat number : ");**

**seatnum = scanner.nextInt();**

**correct = false;**

**//validation**

**if (seatnum<0 || seatnum >= array[rownum].length) {**

**System.out.println("Invaild seat number!");**

**correct = true;**

**}**

**} catch(Exception x){**

**System.out.println("Invaild seat number!");**

**scanner.next(); //Clear the buffer**

**}**

**}**

**if (array[rownum][seatnum-1] == 0) {**

**System.out.println("\nSeat is already Available.");**

**} else{**

**array[rownum][seatnum-1] = 0;**

**//tickets array to find and cancel the corresponding ticket**

**for(int i = 0; i < tickets.length; i++){**

**if (tickets[i] != null && tickets[i].getRow() == rowletter && tickets[i].getSeat() == seatnum) {**

**tickets[i] = null; //Remove the ticket**

**break;**

**}**

**}**

**System.out.println("\nSeat Successfully Canceled.");**

**}**

**}**

**//Find the first available seat in the array**

**private static void find\_first\_available(){**

**for(int i = 0; i < array.length; i++){**

**for(int j = 0; j < array[i].length; j++){**

**if (array[i][j] == 0) {**

**char row = (char) ('A' + i);**

**System.out.println("\nFirst available seat is: " + row + (j + 1));**

**return; //return the value when the first available seat is found**

**}**

**}**

**}**

**}**

**//Method to shows the seats that are available(O) and the seats that have been sold(X)**

**private static void show\_seating\_plan(){**

**for (int i = 0; i < array.length; i++) {**

**for (int j = 0; j < array[i].length; j++) {**

**if (array[i][j] == 0) {**

**System.out.print("O "); //represent an available seat**

**} else {**

**System.out.print("X "); //represent a sold seat**

**}**

**}**

**System.out.println(); //Move to the next line after printing seats of the current row**

**}**

**}**

**//Method to prints the information of all tickets that have been sold**

**public static void print\_tickets\_info(){**

**int total = 0;**

**for(Ticket element: tickets){**

**if (element != null) {**

**element.printInfo();**

**total += element.getPrice();**

**System.out.println();**

**}**

**}**

**System.out.println("Total Price: £" + total); //Print the total price of the tickets sold**

**}**

**//Method to search the ticket from person's information**

**private static void search\_ticket(Scanner scanner){**

**//asks the user to input a row letter**

**System.out.print("Enter row letter (A,B,C or D): ");**

**char rowletter = scanner.next().toUpperCase().charAt(0);**

**int rownum = (int) (rowletter - 'A');**

**//validation**

**if (rownum < 0 || rownum >= array.length) {**

**System.out.println("Invalid row letter!");**

**search\_ticket(scanner); //Call to allow the user to input the row letter again**

**return;**

**}**

**int seatnum = 0;**

**boolean correct = true;**

**while (correct) {**

**try{**

**//asks the user to input a seat number**

**System.out.print("Enter seat number : ");**

**seatnum = scanner.nextInt();**

**correct = false;**

**//validation**

**if (seatnum<0 || seatnum >= array[rownum].length) {**

**System.out.println("Invaild seat number!");**

**correct = true;**

**}**

**} catch(Exception x){**

**System.out.println("Invaild seat number!");**

**scanner.next(); //Clear the buffer**

**}**

**}**

**if (array[rownum][seatnum-1] == 0) {**

**System.out.println("\nThis seat is Available! ");**

**} else{**

**//Search for the ticket corresponding to the provided row letter and seat number**

**for(int i = 0; i < tickets.length; i++){**

**if (tickets[i].getRow() == rowletter && tickets[i].getSeat() == seatnum){**

**tickets[i].printInfo(); //Print ticket & person informations**

**break;**

**}**

**}**

**}**

**}**

**}**

**Person.java**

**public class Person {**

**private String firstname;**

**private String surname;**

**private String email;**

**//Contructor to initialize Person object**

**public Person(String firstname, String surname, String email){**

**this.firstname = firstname;**

**this.surname = surname;**

**this.email = email;**

**}**

**//getter methods**

**public String getfirstname(){**

**return firstname;**

**}**

**public String getsurname(){**

**return surname;**

**}**

**public String getemail(){**

**return email;**

**}**

**//method to print person informations**

**public void printperson(){**

**System.out.println("Full Name: "+ firstname +" "+ surname);**

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

**}**

**}**

**Ticket.java**

**import java.io.File; //Import the File class**

**import java.io.FileWriter; //Import the filewriter class**

**import java.io.IOException; //Import the IOException class to handle errors**

**public class Ticket {**

**private char row;**

**private int seat;**

**public double price;**

**private Person person;**

**//Contructor to initialize Ticket object**

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

**this.row = row;**

**this.seat = seat;**

**this.price = price;**

**this.person = person;**

**}**

**//Getter methods**

**public char getRow() {**

**return row;**

**}**

**public int getSeat() {**

**return seat;**

**}**

**public double getPrice() {**

**return price;**

**}**

**public Person getPerson() {**

**return person;**

**}**

**//Method to print ticket informations**

**public void printInfo() {**

**System.out.println();**

**System.out.println("Row: " + row + "\t Seat: " + seat);**

**System.out.println("Price: £" + price );**

**System.out.println("Personal Details -");**

**person.printperson(); //Method in the Person class to print person details**

**}**

**//Method to save ticket information to files**

**public void save(){**

**String FileName = String.valueOf(row) + String.valueOf(seat) + ".txt"; //Constructing File name**

**File myObj = new File(FileName); //Creating a File object**

**try {**

**FileWriter myWriter = new FileWriter(FileName); //Creating FileWriter object to write to the File**

**myWriter.write("Row: "+ row + "\t Seat: " + seat + "\n");**

**myWriter.write("Price: £" + price + "\n");**

**myWriter.write("Personal Details - "+ "\n");**

**myWriter.write("Full Name: "+ person.getfirstname() + "" + person.getsurname() + "\n");**

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

**myWriter.close();**

**System.out.println("Ticket information saved to " + FileName);**

**} catch (IOException e) {**

**System.out.println("An error occurred while saving ticket Information! ");**

**e.printStackTrace(); //Return some details about error**

**}**

**}**

**}**

<<END>>