Informatics Institute of Technology

School of Computing

Software Development II Coursework Report

Module	: 4COSC010C.2: Software Development II (2023)
Date of submission	: 22/03/2024
Student ID	: 20221813/ w2053245
Student First Name	: Neth
Student Surname	: Botheju

Tutorial group (day, time, and tutor/s): G 23 (Wednesday, 8.30a.m - 10.30a.m, Dinusha Ruwan Kumara / Kalhari Walawage)

"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 : Sandalindu Neth Pehesara Botheju

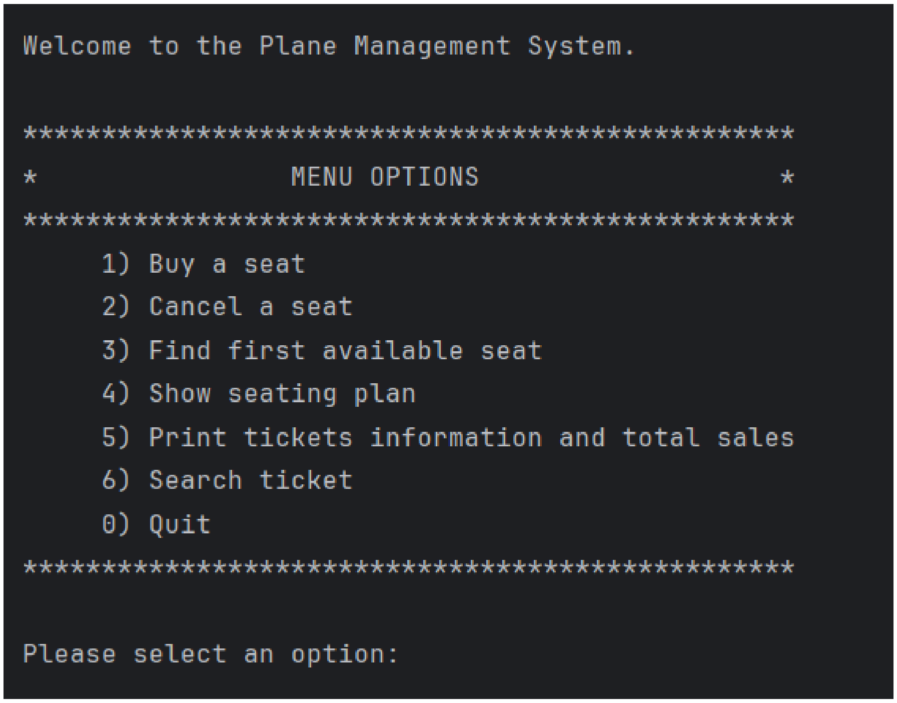
Student ID : 20221813/ w2053245

Self-assessment form and test plan

1) Self-assessment form

Task	**Self-assessment (select one)**	**Comments**
1	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented the welcome message display and initialized the seat management system using 2D arrays.
2	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented user menu with option zero.

Insert here a screenshot of your welcome message and menu:

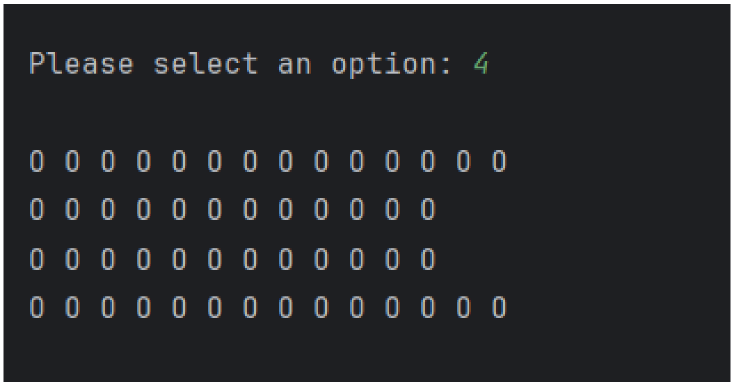


```
Welcome to the Plane Management System.

**************************************************
*                 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:
```

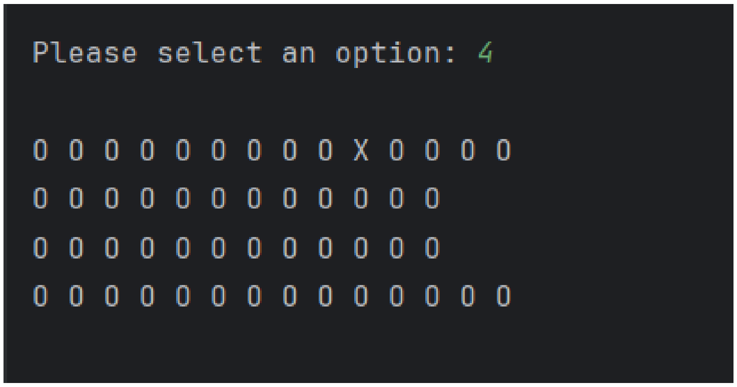
3	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented buy_seat method with user input, check seat availability and record the seat as sold. Call it in menu option 1.
4	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented cancel_seat method with user input, check seat unavailability and record the seat as unsold. Call it in menu option 2.

5	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented find_first_available method which finds first available seat. Call it in menu option 3.
6	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented show_seating_plan method which display available seats with the character 'O' and the sold seats with 'X'. Call it in menu option 4.

Insert here a screenshot of the seating plan:



```
Please select an option: 4

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
```



```
Please select an option: 4

O O O O O O O O O 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
```

7	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented Person class with attributes, constructor, all the getters and setters and print information method.
8	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented Ticket class with attributes, constructor, all the getters and setters and print information method (including the information of the Person).

9	☒Fully implemented ☐Partially implemented ☐Not attempted	Extended buy_seat and cancel_seat methods, when a seat sold it create person and ticket objects and ticket object will add to the ticket ArrayList. If seat unreserved that ticket will remove from ticket ArrayList.
10	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented print_ticket_info method which prints the total amount with slod tickets. Call it in menu option 5.
11	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented search_ticket method which takes seat as input and if it sold print person details. Call it in menu option 6.
12	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented save method in class Ticket which save text file with ticket information (including person details).

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**
Display option menu	Start the program.	Display option menu	As Expected,	☒Pass ☐Fail
Buying a seat	Option 1 in menu. Enter row letter and seat number.	That seat has been successfully booked.	As Expected,	☒Pass ☐Fail
Cancel seat	Option 2 in menu. Enter row letter and seat number.	That seat has been successfully cancelled.	As Expected,	☒Pass ☐Fail
Finding the first available seat	Option 3 in menu.	That will find first available seat in the plane.	As Expected,	☒Pass ☐Fail

Show seat plan	Option 4 in menu.	Display seat plan using ‘O’ and ’X’.	As Expected,	☒Pass ☐Fail
Menu option input validation	Input 8 in option menu.	Display message: Please enter valid option.	As Expected,	☒Pass ☐Fail
Row letter input validation	Input W for the row letter.	Display message: Please enter valid row letter (A-D).	As Expected,	☒Pass ☐Fail

Part B testing

Test case / scenario	Input	Expected Output	Output	Pass/Fail
Print tickets information	Option 5 in menu.	Display total amount with sold tickets.	As Expected,	☒Pass ☐Fail
Search ticket	Option 6 in menu. Enter row letter and seat number.	If that ticket has sold then it will display ticket information.	As Expected,	☒Pass ☐Fail
Personal details for the ticket	Option 1 in menu. Enter row letter and seat number.	when a ticket is sold, it will ask user to input first name, surname, email .	As Expected,	☒Pass ☐Fail
Add ticket to tickets ArrayList	Option 1 in menu. Enter row letter and seat number.	When a ticket is sold, it will create new ticket object and add to tickets ArrayList.	As Expected,	☒Pass ☐Fail
Remove ticket from tickets ArrayList	Option 2 in menu. Enter row letter and seat number.	When a ticket is cancelled, it will remove that ticket from the tickets ArrayList.	As Expected,	☒Pass ☐Fail
Save ticket as text file	Option 1 in menu. Enter row	When a ticket is sold, it will save	As Expected,	☒Pass ☐Fail

	letter and seat number.	ticket details as text file.		
Delete ticket text file	Option 2 in menu. Enter row letter and seat number.	When a ticket is cancelled, it will delete that ticket text file.	As Expected,	☒Pass ☐Fail

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

The recent updates have made the program better at managing data and interacting with users. These are some significant improvements,

- When saving the ticket text file, the program will first check for a folder named "Tickets." If the folder doesn't exist, it will be created. Then, the ticket file will be saved within the "Tickets" folder.
- When a user cancels a seat, the related ticket text file will be deleted.

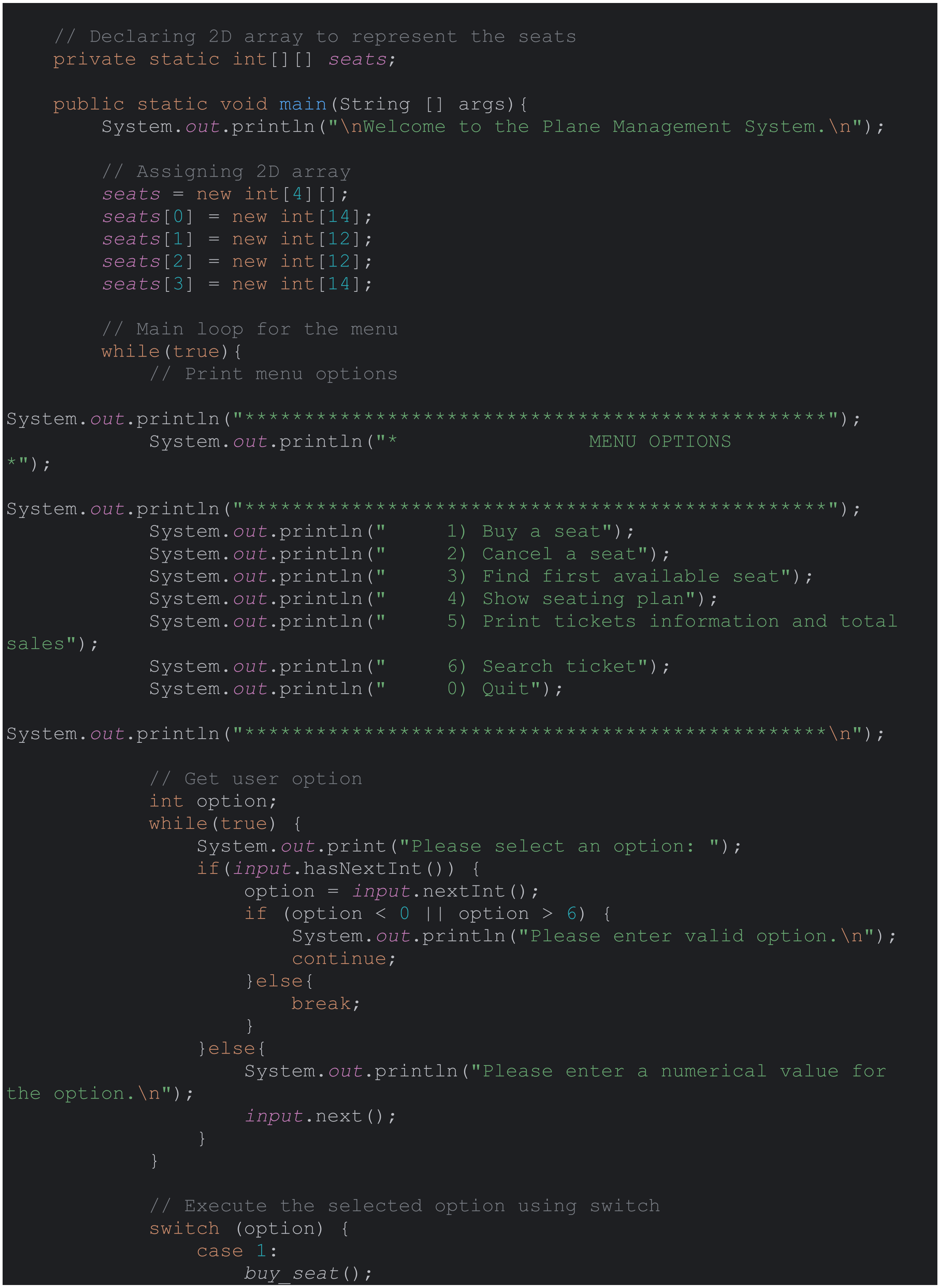
You will need to demonstrate your understanding of the submitted code. Your tutor will arrange a coursework demonstration. During the coursework demonstration, your tutor will ask you to execute your program and questions on your code.

Failure to attend the demonstration will result in 0 for the coursework.

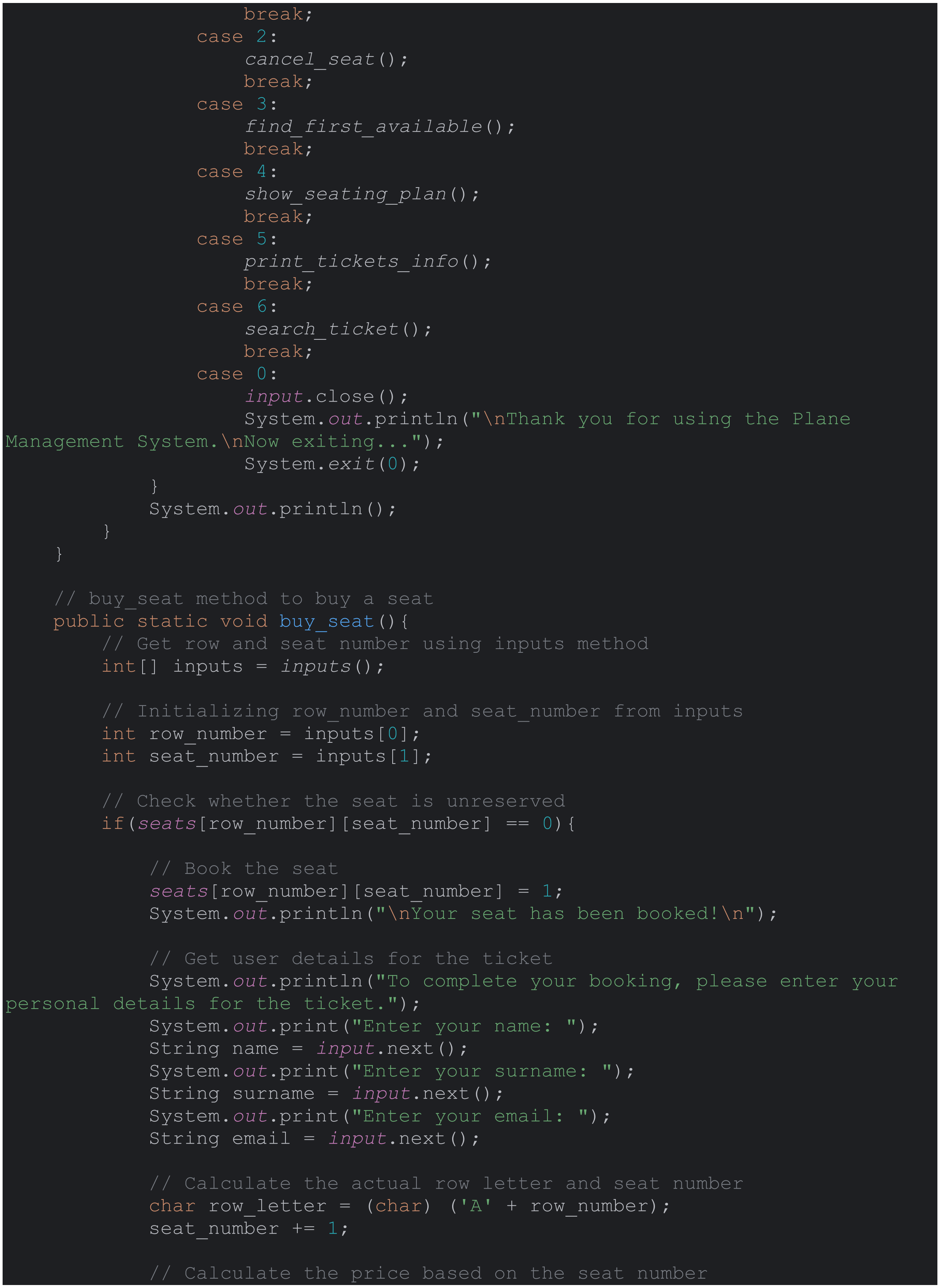
3) Code :

w2053245_PlaneManagement.java

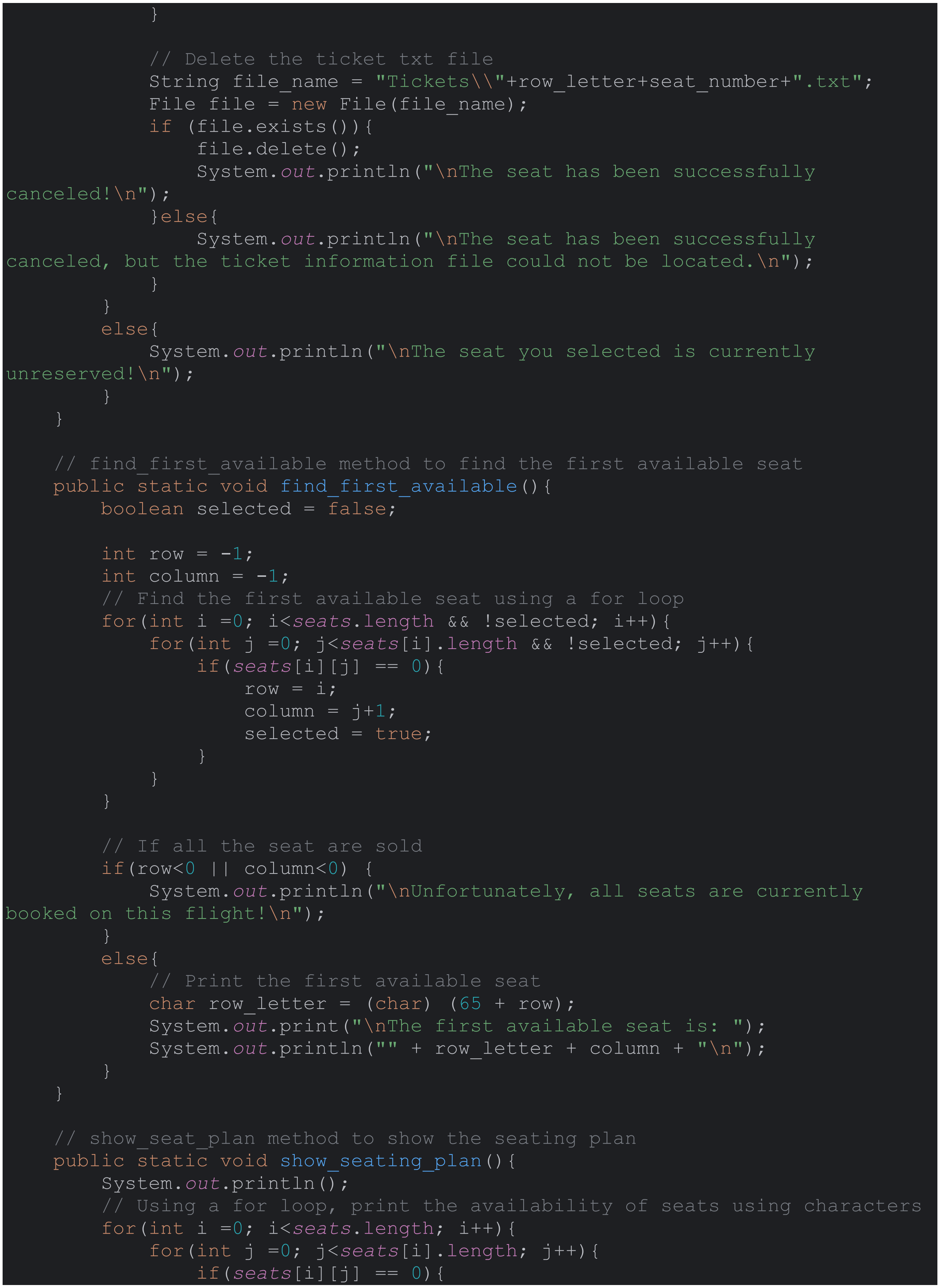
```
import java.util.*;
import java.io.File;

public class w2053245_PlaneManagement {

    // Scanner object for user inputs
    private static Scanner input = new Scanner(System.in);

    // Create ArrayList to store Ticket objects
    private static ArrayList<Ticket> tickets = new ArrayList<Ticket>();
```



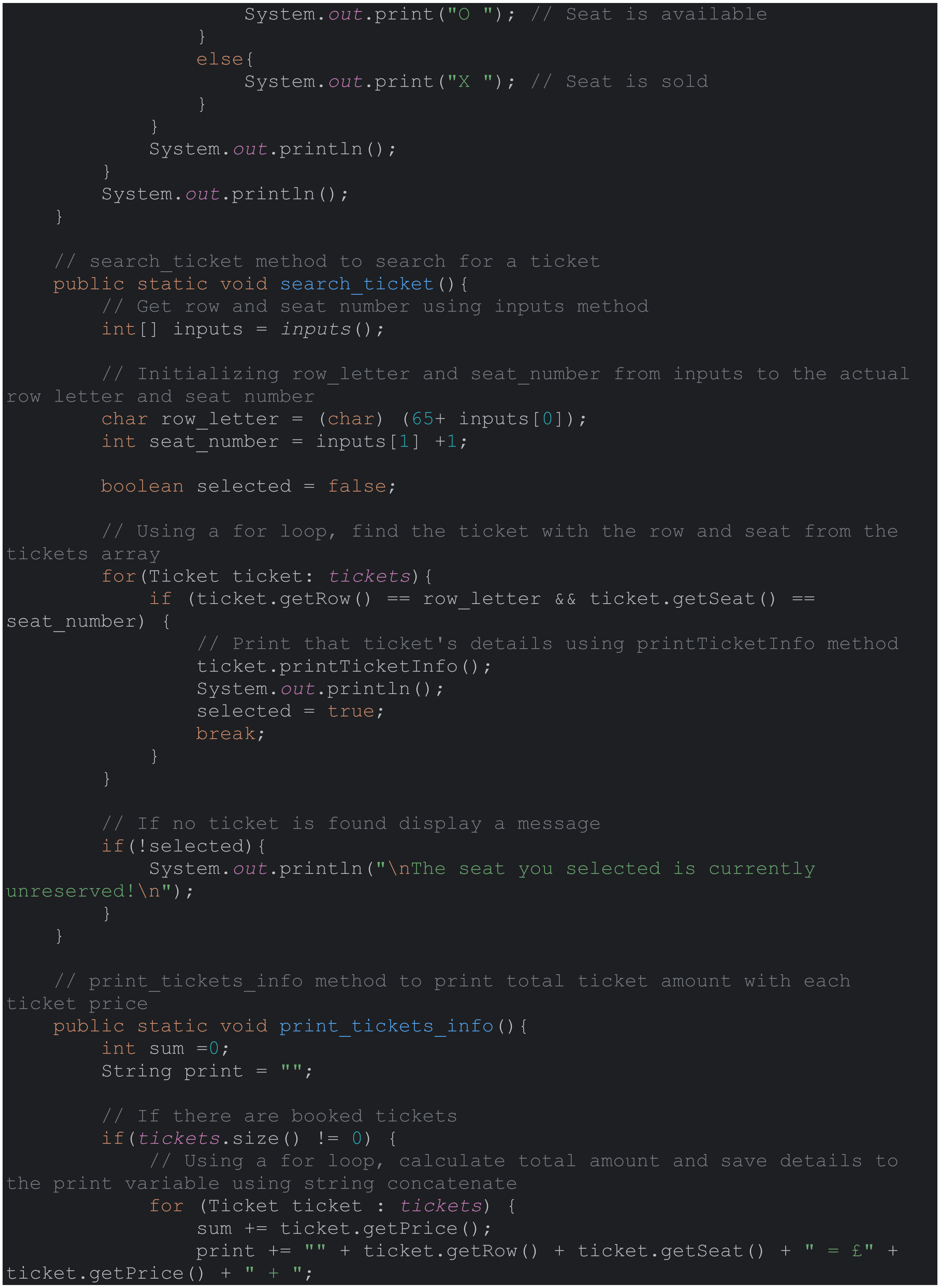
```
    // Declaring 2D array to represent the seats
    private static int[][] seats;

    public static void main(String [] args){
        System.out.println("\nWelcome to the Plane Management System.\n");

        // Assigning 2D array
        seats = new int[4][];
        seats[0] = new int[14];
        seats[1] = new int[12];
        seats[2] = new int[12];
        seats[3] = new int[14];

        // Main loop for the menu
        while(true){
            // Print menu options

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

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("**************************************************\n");

            // Get user option
            int option;
            while(true) {
                System.out.print("Please select an option: ");
                if(input.hasNextInt()) {
                    option = input.nextInt();
                    if (option < 0 || option > 6) {
                        System.out.println("Please enter valid option.\n");
                        continue;
                    }else{
                        break;
                    }
                }else{
                    System.out.println("Please enter a numerical value for
the option.\n");
                    input.next();
                }
            }

            // Execute the selected option using switch
            switch (option) {
                case 1:
                    buy_seat();
```



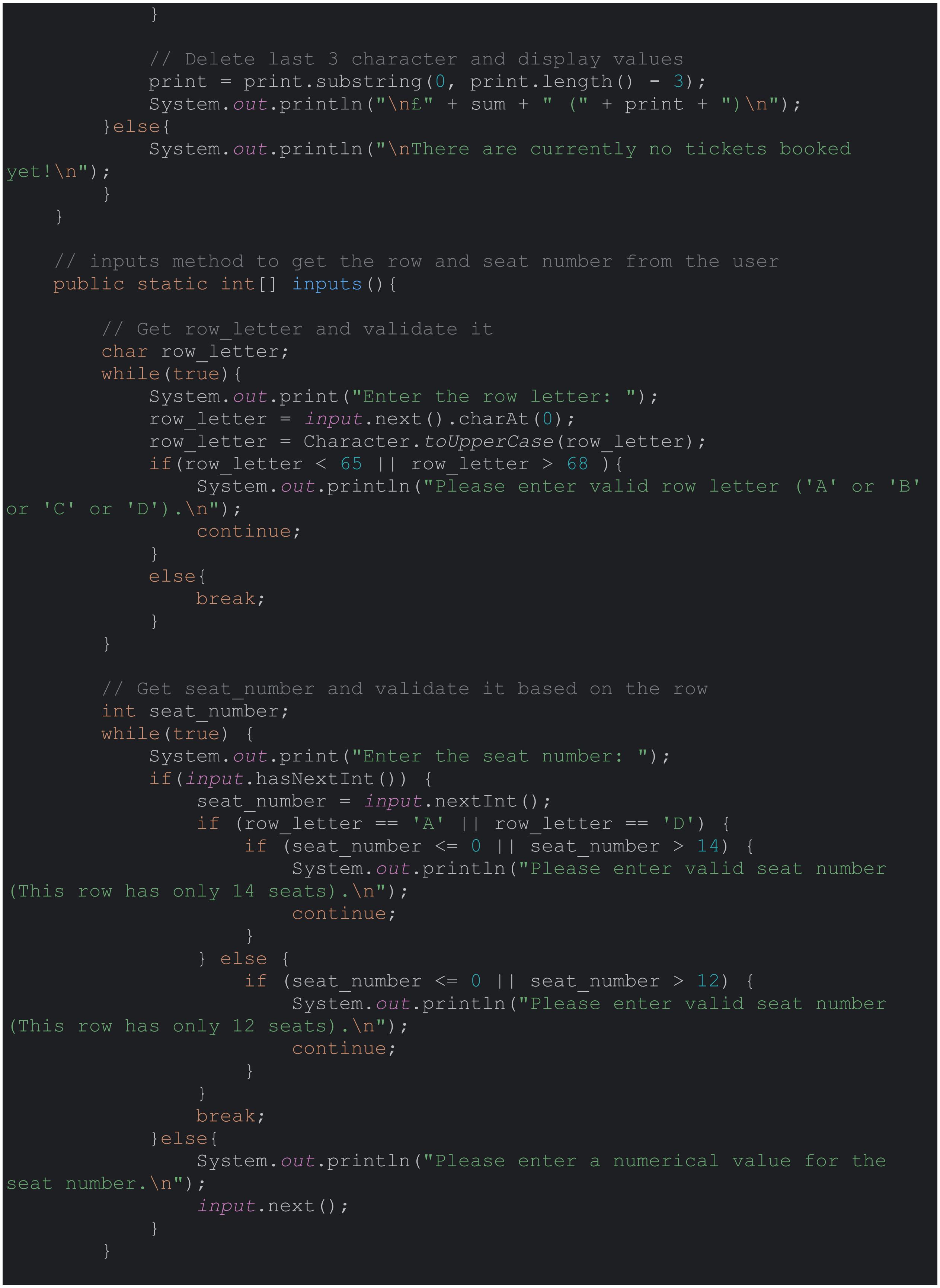
```
                    break;
                case 2:
                    cancel_seat();
                    break;
                case 3:
                    find_first_available();
                    break;
                case 4:
                    show_seating_plan();
                    break;
                case 5:
                    print_tickets_info();
                    break;
                case 6:
                    search_ticket();
                    break;
                case 0:
                    input.close();
                    System.out.println("\nThank you for using the Plane
Management System.\nNow exiting...");
                    System.exit(0);
            }
            System.out.println();
        }
    }

    // buy_seat method to buy a seat
    public static void buy_seat(){
        // Get row and seat number using inputs method
        int[] inputs = inputs();

        // Initializing row_number and seat_number from inputs
        int row_number = inputs[0];
        int seat_number = inputs[1];

        // Check whether the seat is unreserved
        if(seats[row_number][seat_number] == 0){

            // Book the seat
            seats[row_number][seat_number] = 1;
            System.out.println("\nYour seat has been booked!\n");

            // Get user details for the ticket
            System.out.println("To complete your booking, please enter your
personal details for the ticket.");
            System.out.print("Enter your name: ");
            String name = input.next();
            System.out.print("Enter your surname: ");
            String surname = input.next();
            System.out.print("Enter your email: ");
            String email = input.next();

            // Calculate the actual row letter and seat number
            char row_letter = (char) ('A' + row_number);
            seat_number += 1;

            // Calculate the price based on the seat number
            int price;
            if(seat_number<6){
                price = 200;
            }
            else if(seat_number<10){
                price = 150;
            }
            else{
                price = 180;
            }

            // Create person and ticket objects
            Person person = new Person(name, surname, email);
            Ticket ticket =  new Ticket(row_letter, seat_number, price,
person);

            // Add that ticket object to the tickets list
            tickets.add(ticket);

            // Call save method in ticket class to save the ticket
information as a text file
            ticket.save();

            System.out.println("\nYour ticket has been successfully booked
and saved!\n");
        }
        else{
            System.out.println("\nSorry, this seat is already sold. Please
try selecting another seat.\n");
        }
    }

    // cancel_seat method to cancel a seat
    public static void cancel_seat(){
        // Get row and seat number using inputs method
        int[] inputs = inputs();

        // Initializing row_number and seat_number from inputs
        int row_number = inputs[0];
        int seat_number = inputs[1];

        // Check whether the seat is reserved
        if(seats[row_number][seat_number] == 1){
            // Cancel the seat
            seats[row_number][seat_number] = 0;

            // Calculate the actual row letter and seat number
            char row_letter = (char) ('A' + row_number);
            seat_number += 1;

            // Remove that ticket from the tickets list
            for (Ticket ticket : tickets) {
                if (ticket.getRow() == row_letter && ticket.getSeat() ==
seat_number) {
                    tickets.remove(ticket);
                    break;
                }
```



```
            }

            // Delete the ticket txt file
            String file_name = "Tickets\\"+row_letter+seat_number+".txt";
            File file = new File(file_name);
            if (file.exists()){
                file.delete();
                System.out.println("\nThe seat has been successfully
canceled!\n");
            }else{
                System.out.println("\nThe seat has been successfully
canceled, but the ticket information file could not be located.\n");
            }
        }
        else{
            System.out.println("\nThe seat you selected is currently
unreserved!\n");
        }
    }

    // find_first_available method to find the first available seat
    public static void find_first_available(){
        boolean selected = false;

        int row = -1;
        int column = -1;
        // Find the first available seat using a for loop
        for(int i =0; i<seats.length && !selected; i++){
            for(int j =0; j<seats[i].length && !selected; j++){
                if(seats[i][j] == 0){
                    row = i;
                    column = j+1;
                    selected = true;
                }
            }
        }

        // If all the seat are sold
        if(row<0 || column<0) {
            System.out.println("\nUnfortunately, all seats are currently
booked on this flight!\n");
        }
        else{
            // Print the first available seat
            char row_letter = (char) (65 + row);
            System.out.print("\nThe first available seat is: ");
            System.out.println("" + row_letter + column + "\n");
        }
    }

    // show_seat_plan method to show the seating plan
    public static void show_seating_plan(){
        System.out.println();
        // Using a for loop, print the availability of seats using characters
        for(int i =0; i<seats.length; i++){
            for(int j =0; j<seats[i].length; j++){
                if(seats[i][j] == 0){
```



```
                    System.out.print("O "); // Seat is available
                }
                else{
                    System.out.print("X "); // Seat is sold
                }
            }
            System.out.println();
        }
        System.out.println();
    }

    // search_ticket method to search for a ticket
    public static void search_ticket(){
        // Get row and seat number using inputs method
        int[] inputs = inputs();

        // Initializing row_letter and seat_number from inputs to the actual
row letter and seat number
        char row_letter = (char) (65+ inputs[0]);
        int seat_number = inputs[1] +1;

        boolean selected = false;

        // Using a for loop, find the ticket with the row and seat from the
tickets array
        for(Ticket ticket: tickets){
            if (ticket.getRow() == row_letter && ticket.getSeat() ==
seat_number) {
                // Print that ticket's details using printTicketInfo method
                ticket.printTicketInfo();
                System.out.println();
                selected = true;
                break;
            }
        }

        // If no ticket is found display a message
        if(!selected){
            System.out.println("\nThe seat you selected is currently
unreserved!\n");
        }
    }

    // print_tickets_info method to print total ticket amount with each
ticket price
    public static void print_tickets_info(){
        int sum =0;
        String print = "";

        // If there are booked tickets
        if(tickets.size() != 0) {
            // Using a for loop, calculate total amount and save details to
the print variable using string concatenate
            for (Ticket ticket : tickets) {
                sum += ticket.getPrice();
                print += "" + ticket.getRow() + ticket.getSeat() + " = £" +
ticket.getPrice() + " + ";
```



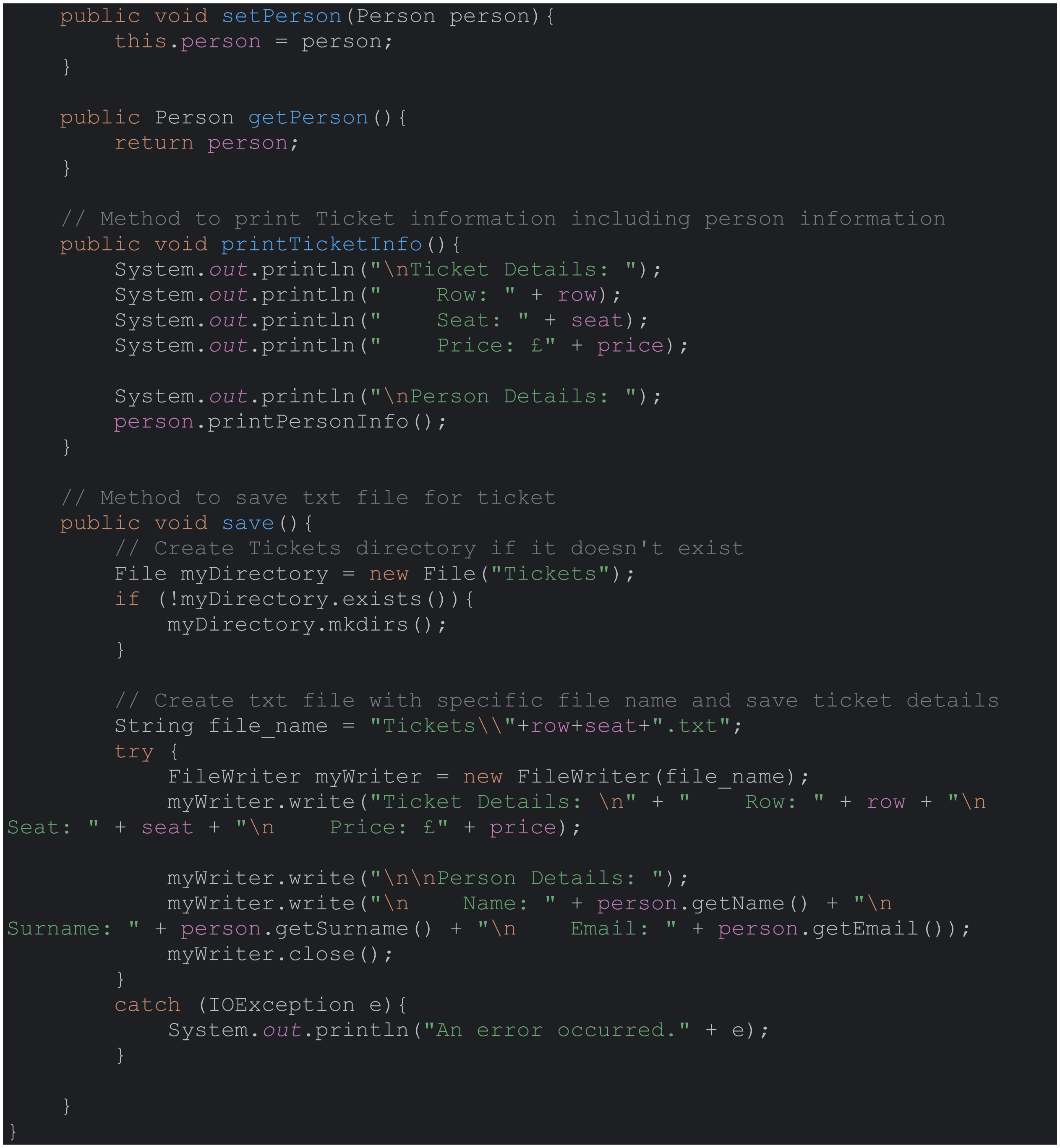
```
            }

            // Delete last 3 character and display values
            print = print.substring(0, print.length() - 3);
            System.out.println("\n£" + sum + " (" + print + ")\n");
        }else{
            System.out.println("\nThere are currently no tickets booked 
yet!\n");
        }
    }

    // inputs method to get the row and seat number from the user
    public static int[] inputs(){

        // Get row_letter and validate it
        char row_letter;
        while(true){
            System.out.print("Enter the row letter: ");
            row_letter = input.next().charAt(0);
            row_letter = Character.toUpperCase(row_letter);
            if(row_letter < 65 || row_letter > 68 ){
                System.out.println("Please enter valid row letter ('A' or 'B' 
or 'C' or 'D').\n");
                continue;
            }
            else{
                break;
            }
        }

        // Get seat_number and validate it based on the row
        int seat_number;
        while(true) {
            System.out.print("Enter the seat number: ");
            if(input.hasNextInt()) {
                seat_number = input.nextInt();
                if (row_letter == 'A' || row_letter == 'D') {
                    if (seat_number <= 0 || seat_number > 14) {
                        System.out.println("Please enter valid seat number 
(This row has only 14 seats).\n");
                        continue;
                    }
                } else {
                    if (seat_number <= 0 || seat_number > 12) {
                        System.out.println("Please enter valid seat number 
(This row has only 12 seats).\n");
                        continue;
                    }
                }
                break;
            }else{
                System.out.println("Please enter a numerical value for the 
seat number.\n");
                input.next();
            }
        }
        // Create int array to return row_letter and seat_number
        int[] return_values = new int[2];

        // Assigning the system side row number and seat number to the array
        return_values[0] = row_letter - 65;
        return_values[1] = seat_number -1;

        return return_values;
    }
}
```

Person.java

```
public class Person {

    // Attributes
    private String name;
    private String surname;
    private String email;

    // Constructor
    public Person(String name, String surname, String email){
        this.name = name;
        this.surname = surname;
        this.email = email;
    }

    // Getters and Setters
    public void setName(String name){
        this.name = name;
    }

    public String getName(){
        return name;
    }

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

    public String getSurname(){
        return surname;
    }

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

    public String getEmail(){
        return email;
    }

    // Method to print Person information
    public void printPersonInfo(){
        System.out.println("    Name: " + name);
        System.out.println("    Surname: " + surname);
        System.out.println("    Email: " + email);
    }
}
```

Ticket.java

```
import java.io.*;

public class Ticket {

    // Attributes
    private char row;
    private int seat;
    private int price;
    private Person person;

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

    // Getters and Setters
    public void setRow(char row){
        this.row = row;
    }

    public char getRow(){
        return row;
    }

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

    public int getSeat(){
        return seat;
    }

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

    public int getPrice(){
        return price;
    }
```



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

    public Person getPerson(){
        return person;
    }

    // Method to print Ticket information including person information
    public void printTicketInfo(){
        System.out.println("\nTicket Details: ");
        System.out.println("    Row: " + row);
        System.out.println("    Seat: " + seat);
        System.out.println("    Price: £" + price);

        System.out.println("\nPerson Details: ");
        person.printPersonInfo();
    }

    // Method to save txt file for ticket
    public void save(){
        // Create Tickets directory if it doesn't exist
        File myDirectory = new File("Tickets");
        if (!myDirectory.exists()){
            myDirectory.mkdirs();
        }

        // Create txt file with specific file name and save ticket details
        String file_name = "Tickets\\"+row+seat+".txt";
        try {
            FileWriter myWriter = new FileWriter(file_name);
            myWriter.write("Ticket Details: \n" + "    Row: " + row + "\n
Seat: " + seat + "\n    Price: £" + price);

            myWriter.write("\n\nPerson Details: ");
            myWriter.write("\n    Name: " + person.getName() + "\n
Surname: " + person.getSurname() + "\n    Email: " + person.getEmail());
            myWriter.close();
        }
        catch (IOException e){
            System.out.println("An error occurred." + e);
        }

    }
}
```