Informatics Institute of Technology

School of Computing

Software Development II Coursework Report

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

Date of submission : 7/15/2024

Student ID : 20230671 / w2054640

Student First Name : Tavishi

Student Surname : Balachandra

Tutorial group (day, time, and tutor/s): Group 21,

Tuesday (10.30am-12.30pm)

Miss. Vishmi Embuldeniya

"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 : Tavishi Balachandra

Student ID : 20230671

Self-assessment form and test plan

Self-assessment form

Task	Self-assessment (select one)	Comments
1	☒Fully implemented ☐Partially implemented ☐Not attempted	The welcome message "Welcome to The London Lumiere" is displayed when the program starts” The seats array is initialized with 48 seats, all set to 0
2	☒Fully implemented ☐Partially implemented ☐Not attempted	The user menu is implemented with options 1 to 8. Input validation ensures the menu repeats until a valid option is selected or the user chooses to exit.

Insert here a screenshot of your welcome message and menu:

```
Hi, Welcome to The London Lumiere

1. Buy a ticket
2. Cancel a ticket
3. Print seating plan
4. Find first available seat
5. Print tickets information
6. Search for a ticket
7. Sort tickets by price
8. Exit

Please select an option:
```

3	☒Fully implemented ☐Partially implemented ☐Not attempted	The ‘buy_ticket’ method checks if the seat number is valid and available, books the seat if it is available, and displays appropriate messages.
4	☒Fully implemented ☐Partially implemented ☐Not attempted	‘cancel_ticket’ method successfully marks a seat as available if it is currently unavailable.Appropriate messages are displayed based on whether the seat was already available or has been successfully cancelled.

Insert here a screenshot of the cancel ticket method

```
Hi, Welcome to The London Lumiere

1. Buy a ticket
2. Cancel a ticket
3. Print seating plan
4. Find first available seat
5. Print tickets information
6. Search for a ticket
7. Sort tickets by price
8. Exit

Please select an option:2
Enter row number (1-3): 1
Enter seat number (1-16): 5
This seat is already available
```

```
The seat has been booked

1. Buy a ticket
2. Cancel a ticket
3. Print seating plan
4. Find first available seat
5. Print tickets information
6. Search for a ticket
7. Sort tickets by price
8. Exit

Please select an option:2
Enter row number (1-3): 2
Enter seat number (1-16): 5
The seat has been cancelled
```

5	☒Fully implemented ☐Partially implemented ☐Not attempted	Display available seats with the character ‘O’ and the sold seats with ‘X’

Insert here a screenshot of the print seating area

```
Please select an option:3

Seating area (O = available, X = sold):

*************************
*        SCREEN         *
*************************

OOOOOOOO  XOOOOOOO  ($12)
OOOOOXOO  OOOOOOOO  ($10)
OOOOOOOO  OOOOOOOO  ($8)
```

6	☒Fully implemented ☐Partially implemented ☐Not attempted	It wii be find the first seat which is still available.
7	☒Fully implemented ☐Partially implemented ☐Not attempted	This method that prints the information from Person.
8	☒Fully implemented ☐Partially implemented ☐Not attempted	This method that prints the information of a Ticket (including the information of the Person).
9	☒Fully implemented ☐Partially implemented ☐Not attempted	Implemented buy_ticket to add a new ticket with person details to the tickets array. Implemented cancel_ticket to remove a ticket from the tickets array when cancelled.
10	☒Fully implemented ☐Partially implemented	This method prints the information of all

	☐Not attempted	tickets that have been sold during the session (including the person's details), and calculates the total price of the tickets sold during the session.
11	☒Fully implemented ☐Partially implemented ☐Not attempted	This method that asks the user to input a row and seat numbers and checks if someone has bought that seat. If someone has bought the seat, use a search algorithm to search the ticket and print the Ticket and Person information; otherwise, it should display 'This seat is available'.
12	☒Fully implemented ☐Partially implemented ☐Not attempted	This method that uses a sorting algorithm to sort the array of Tickets by price and prints all the tickets information (ascending order)

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 ticket	Enter row number (1-3): 1 Enter seat number (1-16): 5 Enter your name: Tavishi Enter your surname: Balachandra Enter your email: tavishi13balachandra@gmail.com	The seat has been booked	The seat has been booked	☒Pass ☐Fail
Cancel a ticket	Enter row number (1-3): 1 Enter seat number (1-16): 5	The seat has been cancelled	The seat has been cancelled	☒Pass ☐Fail
Cancel a ticket	Enter row number (1-3): 2 Enter seat number (1-16): 2	This seat is already available	This seat is already available	☒Pass ☐Fail
Buy a ticket	Enter row number (1-3): 2 Enter seat number (1-16): 5 Enter your name: Tavishi Enter your surname: Balachandra Enter your email: tavishi13balachandra@gmail.com	The seat has been booked	The seat has been booked	☒Pass ☐Fail
Print seating area		Only 2nd Row Seat 5 being marked X	Only 2nd Row Seat 5 being marked X	☒Pass ☐Fail
Find first available seat		Row 1, Seat 1	Row 1, Seat 1	☒Pass ☐Fail
Buy a ticket	Please select an option:1 Enter row number (1-3): 1 Enter seat number (1-16): 1 Enter your name: Tavishi Enter your surname: Parindya Enter your email: parindya@gmail.com	The seat has been booked	The seat has been booked	☒Pass ☐Fail

Enter invalid email	Enter your email: parindyagmail.com	Invalid email. Please enter a valid email address	Invalid email. Please enter a valid email address	☒Pass ☐Fail
Enter the invalid row selection	5	Invalid row number. Please try again.	Invalid row number. Please try again.	☒Pass ☐Fail
Enter the invalid menu option	20	Invalid option. Please try again.	Invalid option. Please try again.	☒Pass ☐Fail
Find first available seat		Row 1, Seat 2	Row 1, Seat 2	☒Pass ☐Fail

Part B testing

Test case / scenario	**Input**	**Expected Output**	**Output**	**Pass/Fail**
Print tickets information		Row: 2, Seat: 5, Price: $10 Name: Tavishi, Surname: Balachandra, Email: tavishi13balachandra@gmail.com Row: 1, Seat: 1, Price: $12 Name: Tavishi, Surname: Parindya, Email: parindya@gmail.com Total price of tickets sold: $22	Row: 2, Seat: 5, Price: $10 Name: Tavishi, Surname: Balachandra, Email: tavishi13balachandra@gmail.com Row: 1, Seat: 1, Price: $12 Name: Tavishi, Surname: Parindya, Email: parindya@gmail.com Total price of tickets sold: $22	☒Pass ☐Fail
Buy a ticket	Enter row number (1-3): 1 Enter seat number (1-16): 1	Seat already booked.	Seat already booked.	☒Pass ☐Fail
Print tickets		Row: 2, Seat: 5, Price: $10	Row: 2, Seat: 5, Price: $10	☒Pass

information		Name: Tavishi, Surname: Balachandra, Email: tavishi13balachandra@gmail.com Row: 1, Seat: 1, Price: $12 Name: Tavishi, Surname: Parindya, Email: parindya@gmail.com Total price of tickets sold: $22	Name: Tavishi, Surname: Balachandra, Email: tavishi13balachandra@gmail.com Row: 1, Seat: 1, Price: $12 Name: Tavishi, Surname: Parindya, Email: parindya@gmail.com Total price of tickets sold: $22	☐Fail
Cancel a ticket	Enter row number (1-3): 1 Enter seat number (1-16): 1	The seat has been cancelled	The seat has been cancelled	☒Pass ☐Fail
Print tickets information		Row: 2, Seat: 5, Price: $10 Name: Tavishi, Surname: Balachandra, Email: tavishi13balachandra@gmail.com Total price of tickets sold: $10	Row: 2, Seat: 5, Price: $10 Name: Tavishi, Surname: Balachandra, Email: tavishi13balachandra@gmail.com Total price of tickets sold: $10	☒Pass ☐Fail
Buy a ticket	Enter row number (1-3): 3 Enter seat number (1-16): 8 Enter your name: Nehara Enter your surname: peris Enter your email: nehara@gmail.com	The seat has been booked	The seat has been booked	☒Pass ☐Fail
Buy a ticket	Enter row number (1-3): 2 Enter seat number (1-16): 4 Enter your name: Helith	The seat has been booked	The seat has been booked	☒Pass ☐Fail

	Enter your surname: nimdinu Enter your email: Helith@gmail.com			
Search for a ticket	Enter row number (1-5): 1 Enter seat number (1-16): 1	This seat is available	This seat is available	☒Pass ☐Fail
Search for a ticket	Enter row number (1-5): 3 Enter seat number (1-16): 8	Row: 3, Seat: 8, Price: $8 Name: Nehara, Surname: peris, Email: nehara@gmail.com	Row: 3, Seat: 8, Price: $8 Name: Nehara, Surname: peris, Email: nehara@gmail.com	☒Pass ☐Fail
Sort tickets by price		Row: 3, Seat: 8, Price: $8 Name: Nehara, Surname: peris, Email: nehara@gmail.com Row: 2, Seat: 5, Price: $10 Name: Tavishi, Surname: Balachandra, Email: tavishi13balachandra@gmail.com Row: 2, Seat: 4, Price: $10 Name: Helith, Surname: nimdinu, Email: Helith@gmail.com	Row: 3, Seat: 8, Price: $8 Name: Nehara, Surname: peris, Email: nehara@gmail.com Row: 2, Seat: 5, Price: $10 Name: Tavishi, Surname: Balachandra, Email: tavishi13balachandra@gmail.com Row: 2, Seat: 4, Price: $10 Name: Helith, Surname: nimdinu, Email: Helith@gmail.com	☒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 :

Main Class

```
import java.util.InputMismatchException;
import java.util.Scanner;

public class CinemaManagement {
    private static final int ROWS = 3;
    private static final int SEATS = 16;
    private static final int[] PRICES = {12, 10, 8}; // Prices for each row
    private static int[][] seats = new int[ROWS][SEATS];
    private static Ticket[] tickets = new Ticket[ROWS * SEATS];
    private static int ticketCount = 0;

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("\nHi, Welcome to The London Lumiere");
        while (true) {
            displayMenu();

            try {
                int choice = scanner.nextInt();
                switch (choice) {
                    case 1:
                        buy_ticket(scanner);
                        break;
                    case 2:
                        cancel_ticket(scanner);
                        break;
                    case 3:
                        print_seating_area();
                        break;
                    case 4:
                        find_first_available();
                        break;
                    case 5:
                        print_tickets_info();
                        break;
                    case 6:
                        search_ticket(scanner);
                        break;
                    case 7:
                        sort_tickets();
                        break;
                    case 8:
                        System.out.println("Exiting program.");
                        return;
                    default:
                        System.out.println("Invalid option. Please try
again.");
                }
            } catch (InputMismatchException inputMismatchException) {
                System.out.println("Invalid input. Please try again.");
                scanner.nextLine(); // Clear the invalid input
```

```
            }
        }
    }

    private static void displayMenu() {
        System.out.format("""

                1. Buy a ticket
                2. Cancel a ticket
                3. Print seating plan
                4. Find first available seat
                5. Print tickets information
                6. Search for a ticket
                7. Sort tickets by price
                8. Exit

                Please select an option: """);
    }

    private static void buy_ticket(Scanner scanner) {
        int row = getValidSeatInput(scanner, "Enter row number (1-3): ",
ROWS) - 1;
        int seat = getValidSeatInput(scanner, "Enter seat number (1-16): ",
SEATS) - 1;

        if (seats[row][seat] == 0) {
            scanner.nextLine();
            String name = getValidName(scanner, "Enter your name: ");
            String surname = getValidName(scanner, "Enter your surname: ");
            String email = getValidEmail(scanner, "Enter your email: ");

            Person person = new Person(name, surname, email);
            int price = PRICES[row];
            Ticket ticket = new Ticket(row, seat, price, person);

            seats[row][seat] = 1;
            tickets[ticketCount++] = ticket;
            System.out.println("The seat has been booked");
        } else {
            System.out.println("Seat already booked.");
        }
    }

    private static String getValidName(Scanner scanner, String prompt) {
        String name;
        while (true) {
            System.out.print(prompt);
            name = scanner.nextLine().trim();
            if (name.matches("[a-zA-Z]+")) {
                return name;
            } else {
                System.out.println("Invalid name. Please enter letters
only.");
            }
        }
    }
```

```
    private static String getValidEmail(Scanner scanner, String prompt) {
        String email;
        while (true) {
            System.out.print(prompt);
            email = scanner.nextLine().trim();
            if (email.contains("@")) {
                return email;
            } else {
                System.out.println("Invalid email. Please enter a valid email
address.");
            }
        }
    }

    private static void cancel_ticket(Scanner scanner) {
        int row = getValidSeatInput(scanner, "Enter row number (1-3): ",
ROWS) - 1;
        int seat = getValidSeatInput(scanner, "Enter seat number (1-16): ",
SEATS) - 1;

        if (seats[row][seat] == 1) {
            seats[row][seat] = 0;
            for (int i = 0; i < ticketCount; i++) {
                if (tickets[i].getRow() == row && tickets[i].getSeat() ==
seat) {
                    tickets[i] = tickets[--ticketCount]; // Remove the ticket
                    tickets[ticketCount] = null; // Nullify the last element
                    System.out.println("The seat has been cancelled");
                    return;
                }
            }
        } else {
            System.out.println("This seat is already available");
        }
    }

    private static void print_seating_area() {
        System.out.println("\nSeating area (O = available, X = sold):");

        System.out.format("""

                *************************
                *        SCREEN         *
                *************************

                """);

        for (int row = 0; row < ROWS; row++) {
            for (int seat = 0; seat < SEATS; seat++) {
                if (seat == 8) System.out.print("  "); // Gap between seats 8
and 9
                System.out.print(seats[row][seat] == 0 ? 'O' : 'X');
            }
            System.out.println("\t" + "($" + PRICES[row] + ")");
        }
    }
```

```
    private static void find_first_available() {
        for (int row = 0; row < ROWS; row++) {
            for (int seat = 0; seat < SEATS; seat++) {
                if (seats[row][seat] == 0) {
                    System.out.println("First available seat: Row " + (row +
1) + ", Seat " + (seat + 1));
                    return;
                }
            }
        }
        System.out.println("No available seats");
    }

    private static void print_tickets_info() {
        int total = 0;
        for (int i = 0; i < ticketCount; i++) {
            tickets[i].printTicketInfo();
            total += tickets[i].getPrice();
        }
        System.out.println("Total price of tickets sold: $" + total);
    }

    private static void search_ticket(Scanner scanner) {
        System.out.println("Enter row number (1-5): ");
        int row = scanner.nextInt() - 1;
        System.out.println("Enter seat number (1-16): ");
        int seat = scanner.nextInt() - 1;

        if (isValidSeat(row, seat) && seats[row][seat] == 1) {
            for (int i = 0; i < ticketCount; i++) {
                if (tickets[i].getRow() == row && tickets[i].getSeat() ==
seat) {
                    tickets[i].printTicketInfo();
                    return;
                }
            }
        } else if (isValidSeat(row, seat) && seats[row][seat] == 0) {
            System.out.println("This seat is available");
        } else {
            System.out.println("Invalid seat number");
        }
    }

    private static void sort_tickets() {
        for (int i = 0; i < ticketCount - 1; i++) {
            for (int j = 0; j < ticketCount - i - 1; j++) {
                if (tickets[j].getPrice() > tickets[j + 1].getPrice()) {
                    Ticket temp = tickets[j];
                    tickets[j] = tickets[j + 1];
                    tickets[j + 1] = temp;
                }
            }
        }
        for (int i = 0; i < ticketCount; i++) {
            tickets[i].printTicketInfo();
        }
    }
```

```
    private static boolean isValidSeat(int row, int seat) {
        return row >= 0 && row < ROWS && seat >= 0 && seat < SEATS;
    }

    private static int getValidSeatInput(Scanner scanner, String prompt, int
max) {
        int input = -1;
        boolean valid = false;
        while (!valid) {
            System.out.print(prompt);
            try {
                input = scanner.nextInt();
                if (input >= 1 && input <= max) {
                    valid = true;
                } else {
                    System.out.println("Invalid seat number. Please try
again.");
                }
            } catch (InputMismatchException e) {
                System.out.println("Invalid input. Please enter an
integer.");
                scanner.next(); // Clear the invalid input
            }
        }
        return input;
    }
}
```

Ticket Class

```
public class Ticket {
    private int row;
    private int seat;
    private int price;
    private Person person;

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

    public int getRow() {
        return row;
    }

    public void setRow(int row) {
        this.row = row;
```

```
    }

    public int getSeat() {
        return seat;
    }

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

    public int getPrice() {
        return price;
    }

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

    public Person getPerson() {
        return person;
    }

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

    public void printTicketInfo() {
        System.out.println("Row: " + (row + 1) + ", Seat: " + (seat + 1) + ",
Price: $" + price);
        person.printPersonInfo();
    }
}
```

Person Class

```
public class Person {
    private String name;
    private String surname;
    private String email;

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

    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;
    }

    public void printPersonInfo() {
        System.out.println("Name: " + name + ", Surname: " + surname + ",
Email: " + email);
    }
}
```

<<END>>