Advanced Programming (COS10033) Assignment 1 Cinema Booking System

Lecturer: Ms. Lushaka Nisansala

Student Name: M.A. Nidula Sanketh Mallikarachchi

Course: UniLink Diploma in IT

Swinburne ID: 104756611

NCHS ID: 2023040050

Email: 104756611@student.swin.edu.au

Contact Number: 0741966164

Cinema Booking System Documentation

Overview

The Cinema Booking System is a Java program designed to manage and facilitate the booking and refunding of seats in a cinema. The program allows users to interact with a cinema hall, view available seats, book seats of different types, and generate receipts. The system also maintains statistics such as total income and the number of seats booked.

Project Structure

The project consists of two Java files:

1. Demo.java: The main class that contains the main method to execute the program. It handles user input, menu options, and calls functions to interact with the Cinema class.
2. Cinema.java: A class representing the Cinema, with methods to initialize the cinema hall, print the cinema layout, book and refund seats, generate receipts, and display statistics.

Classes and Methods

1. Cinema Class

Instance Variables

- row: Number of rows in the cinema hall.
- col: Number of seats per row.
- totalNumberOfSeats: Total number of seats in the cinema hall.
- date: Date of the cinema event.
- sPrice: Price for a standard seat.
- fPrice: Price for a frequent seat.
- pPrice: Price for a pensioner seat.
- cinemaHall: Array representing the status of each seat in the cinema hall.
- totalIncome: Total income generated from booked seats.
- noOfSeatsBooked: Total number of seats currently booked.

Methods

- initializeCinema(): Initializes the cinema hall with empty seats.
- printCinema(): Prints the current status of the cinema hall, marking booked seats with colors.
- isSeatAvailable(int seatNumber): Checks if a specific seat is available for booking.
- bookSeats(int seatNumber, char seatType): Books a seat and updates relevant statistics.
- generateReceipt(...): Generates a booking or refund receipt and prints it.
- refundSeats(int seatNumber): Refunds a booked seat and updates relevant statistics.
- returnSeatType(int seatNumber): Returns the type of a booked seat.
- printStatReport(): Prints a statistics report including the total income and the number of seats booked.

2. Demo Class

- main(String[] args): The main method that serves as the entry point for the program. It handles user input, menu options, and invokes methods from the Cinema class based on user choices.

Code:

Demo.java

```java
package assignment1;

import java.util.Scanner;

public class Demo {
    public static Scanner scan = new Scanner(System.in);

    private static String green = "\u001B[38;2;11;244;102m";
    private static String red = "\u001B[31m";
    private static String magenta = "\u001B[35m";
    private static String endColor = "\u001B[0m";

    public static void main(String[] args) {

        System.out.println(green+"\n========WELCOME TO CINEMA BOOKING
SYSTEM========\n\n"+endColor);

        String date;
        int row, col, totalNumberOfSeats;
        double sPrice, fPrice, pPrice;

        System.out.print("Enter Number of Rows: ");
        row = scan.nextInt();

        col = 10;
        totalNumberOfSeats = row * col;

        scan.nextLine();
        System.out.print("Enter Date: ");
        date = scan.nextLine();

        System.out.print("Enter the Seat Price for Standard:\t $");
        sPrice = scan.nextDouble();

        System.out.print("Enter the Seat Price for Frequent:\t $");
        fPrice = scan.nextDouble();

        System.out.print("Enter the Seat Price for Pensioner:\t $");
        pPrice = scan.nextDouble();

        Cinema cinema = new Cinema(row, col, totalNumberOfSeats, date, sPrice, fPrice,
pPrice);

        while (true) {
            int option;

            System.out.println(magenta+"\n\n----------MENU----------"+endColor);
            System.out.println(green);
            System.out.println("\t1. Display Cinema Seats");
            System.out.println("\t2. Book Seats");
            System.out.println("\t3. Refund Seats");
            System.out.println("\t4. Print Statistics Report");
            System.out.println("\t5. Exit");
            System.out.println(endColor);

            do {
                System.out.print(green+"Enter Option: "+endColor);
                option = scan.nextInt();
                if (option <= 0 || option > 5) {
                    System.out.println("Please Select Valid Option!");
                }
            } while (option <= 0 || option > 5);

            switch (option) {
                case 1:
                    System.out.println(magenta+"\n----------Display Cinema Seats---------
-"+endColor);
                    cinema.printCinema();
                    break;

                case 2:
                    System.out.println(magenta+"\n\n----------Book Cinema Seats----------
\n\n"+endColor);
                    bookSeats(cinema);
                    break;

                case 3:
                    System.out.println(magenta+"\n\n----------Refund Cinema Seats--------
--\n"+endColor);
                    refundSeats(cinema);
                    break;

                case 4:
                    System.out.println(magenta+"\n\n----------Generating Statistics
Report----------\n"+endColor);
                    cinema.printStatReport();
                    break;

                case 5:
                    System.out.println(red+"Program Terminated!"+endColor);
                    System.exit(0);
            }

        }
    }

    private static void bookSeats(Cinema cinema) {
        int totalNumberOfSeatsToBook, bookSeatNumber;
        char seatType, temp;
        double personTotal = 0;

        double standardTotal = 0, frequentTotal = 0, pensionareTotal = 0;
        String standardSeats = "", frequentSeats = "", pensionareSeats = "";
        int standardCount = 0, frequentCount = 0, pensionareCount = 0;

        int numberOfSeatsLeftToBook;
        numberOfSeatsLeftToBook = cinema.getTotalNumberOfSeats() -
cinema.getNoOfSeatsBooked();

        System.out.println(red + "Maximum Number of Seats That can be Booked: " +
numberOfSeatsLeftToBook + endColor +"\n");

        do {
            System.out.print("How Many Seats Do You Want to Book: ");
            totalNumberOfSeatsToBook = scan.nextInt();

            if (totalNumberOfSeatsToBook < 0) {
                System.out.println("Enter a Valid Number!");
            }
            if (totalNumberOfSeatsToBook > numberOfSeatsLeftToBook) {
                System.out.println("Cannot Book This Many Seats!");
            }
        } while (totalNumberOfSeatsToBook < 0 || totalNumberOfSeatsToBook >
numberOfSeatsLeftToBook);

        int i = 0;

        while (i < totalNumberOfSeatsToBook) {

            boolean seatIsAvailable;

            do {
                do {
                    System.out.print("Enter The Seat Number: ");
                    bookSeatNumber = scan.nextInt();
                    if (bookSeatNumber <= 0 || bookSeatNumber >
cinema.getTotalNumberOfSeats()) {
                        System.out.println("Enter Valid Seat Number!");
                    }
                } while (bookSeatNumber <= 0 || bookSeatNumber >
cinema.getTotalNumberOfSeats());

                seatIsAvailable = cinema.isSeatAvailable(bookSeatNumber);

                if (!seatIsAvailable) {
                    System.out.println("This seat is Already Booked!");
                }
            } while (!seatIsAvailable);

            do {
                System.out.print("Enter The Seat Type (S,F,P): ");
                temp = scan.next().charAt(0);
                seatType = Character.toUpperCase(temp);
                if (seatType != 'S' && seatType != 'F' && seatType != 'P') {
                    System.out.println("Enter Valid Seat Type!");
                }
            } while (seatType != 'S' && seatType != 'F' && seatType != 'P');

            cinema.bookSeats(bookSeatNumber, seatType);

            switch (seatType) {
                case 'S':
                    standardTotal += cinema.getsPrice();
                    standardSeats += " " + bookSeatNumber;
                    standardCount++;
                    break;
                case 'F':
                    frequentTotal += cinema.getfPrice();
                    frequentSeats += " " + bookSeatNumber;
                    frequentCount++;
                    break;
                case 'P':
                    pensionareTotal += cinema.getpPrice();
                    pensionareSeats += " " + bookSeatNumber;
                    pensionareCount++;
                    break;
            }
            i++;

        }

        personTotal = standardTotal + frequentTotal + pensionareTotal;

        cinema.setTotalIncomeIncrement(personTotal);

        cinema.generateReceipt(totalNumberOfSeatsToBook, standardTotal, standardSeats,
frequentTotal, frequentSeats,
                pensionareTotal, pensionareSeats, personTotal, standardCount,
frequentCount, pensionareCount, "Booking");
    }

    private static void refundSeats(Cinema cinema) {
        double personTotal = 0;
        double standardTotal = 0, frequentTotal = 0, pensionareTotal = 0;
        String standardSeats = "", frequentSeats = "", pensionareSeats = "";
        int standardCount = 0, frequentCount = 0, pensionareCount = 0;

        int totalNumberOfSeatsToRefund;
        int refundSeatNumber;

        int numberOfBookedSeats = cinema.getNoOfSeatsBooked();
        System.out.println(red+"Maximum number of Tickets that can be refunded: " +
numberOfBookedSeats+endColor+"\n");
        do {
            System.out.print("How many Seats Do you Want to Refund: ");
            totalNumberOfSeatsToRefund = scan.nextInt();

            if (totalNumberOfSeatsToRefund > numberOfBookedSeats) {
                System.out.println("This Many Seats have not been booked!");
            }
            if (totalNumberOfSeatsToRefund <= 0) {
                System.out.println("Enter Valid Number of Seats!");
            }
        } while (totalNumberOfSeatsToRefund > numberOfBookedSeats ||
totalNumberOfSeatsToRefund <= 0);

        int j = 0;
        while (j < totalNumberOfSeatsToRefund) {
            boolean seatIsAvailable;
            do {
                do {
                    System.out.print("Enter Seat Number: ");
                    refundSeatNumber = scan.nextInt();
                    if (refundSeatNumber > cinema.getTotalNumberOfSeats() ||
refundSeatNumber <= 0) {
                        System.out.println("Enter Valid Number of Seats!");
                    }
                } while (refundSeatNumber > cinema.getTotalNumberOfSeats() ||
refundSeatNumber <= 0);

                seatIsAvailable = cinema.isSeatAvailable(refundSeatNumber);

                if (seatIsAvailable) {
                    System.out.println("This Seat is Not Booked before!");
                }

            } while (seatIsAvailable);

            char seatType = cinema.returnSeatType(refundSeatNumber);
            switch (seatType) {
                case 'S':
                    standardTotal += cinema.getsPrice();
                    standardSeats += " " + refundSeatNumber;
                    standardCount++;
                    break;
                case 'F':
                    frequentTotal += cinema.getfPrice();
                    frequentSeats += " " + refundSeatNumber;
                    frequentCount++;
                    break;
                case 'P':
                    pensionareTotal += cinema.getpPrice();
                    pensionareSeats += " " + refundSeatNumber;
                    pensionareCount++;
                    break;
            }
            cinema.refundSeats(refundSeatNumber);
            j++;
        }
        personTotal = standardTotal + frequentTotal + pensionareTotal;

        cinema.setTotalIncomeDecrement(personTotal);

        cinema.generateReceipt(totalNumberOfSeatsToRefund, standardTotal, standardSeats,
frequentTotal, frequentSeats,
                pensionareTotal, pensionareSeats, personTotal, standardCount,
frequentCount, pensionareCount, "Refund");
    }

}
```

Cinema.java

```
package assignment1;

class Cinema {
    // Instance Variables
    private int row;
    private int col;
    private int totalNumberOfSeats;
    private String date;

    private double sPrice;
    private double fPrice;
    private double pPrice;

    private char[] cinemaHall;

    private double totalIncome;
    private int noOfSeatsBooked;

    private String green = "\u001B[38;2;11;244;102m";
    private String red = "\u001B[31m";
    private String blue = "\u001B[94m";
    private String endColor = "\u001B[0m";

    // Constructor
    Cinema(int row, int col, int totalNumberOfSeats, String date, double sPrice, double
fPrice, double pPrice) {
        this.row = row;
```

```java
        this.col = col;
        this.totalNumberOfSeats = totalNumberOfSeats;
        this.date = date;

        this.sPrice = sPrice;
        this.fPrice = fPrice;
        this.pPrice = pPrice;

        this.cinemaHall = new char[totalNumberOfSeats];

        this.totalIncome = 0;
        this.noOfSeatsBooked = 0;
        initializeCinema();
    }

    // Getters
    public double getsPrice() {
        return sPrice;
    }

    public double getfPrice() {
        return fPrice;
    }

    public double getpPrice() {
        return pPrice;
    }

    public int getNoOfSeatsBooked() {
        return noOfSeatsBooked;
    }

    public int getTotalNumberOfSeats() {
        return totalNumberOfSeats;
    }

    // Setters
    public void setTotalIncomeIncrement(double personTotal) {
        this.totalIncome += personTotal;
    }

    public void setTotalIncomeDecrement(double refundTotal) {
        this.totalIncome -= refundTotal;
    }

    // Methods
    public void initializeCinema() {
        for (int i = 0; i < row; i++) {
            for (int j = 0; j < col; j++) {
                int arrayIndex = (i * 10) + j;
                cinemaHall[arrayIndex] = '-';
            }
        }
    }

    public void printCinema() {
        System.out.println();

System.out.println(green+"===============================================================
======================");
        System.out.println("====================================Cinema
Screen====================================");
```

```
System.out.println("=====================================================================
================\n"+endColor);

        int seatNumber = 1;

        for (int i = 0; i < row; i++) {
            for (int j = 0; j < col; j++) {
                if (j == 5) {
                    System.out.print("\t\t");
                }
                int arrayIndex = seatNumber - 1;
                if (cinemaHall[arrayIndex] != '-') {
                    System.out.print(red);
                    System.out.printf("%02d", seatNumber);
                    System.out.print(":" + cinemaHall[arrayIndex] + "\t");
                    System.out.print(endColor);
                } else {
                    System.out.print(green);
                    System.out.printf("%02d", seatNumber);
                    System.out.print(":" + cinemaHall[arrayIndex] + "\t");
                    System.out.print(endColor);
                }

                seatNumber++;
            }
            System.out.println();
        }
    }

    public boolean isSeatAvailable(int seatNumber) {
        int arrayIndex = seatNumber - 1;
        return cinemaHall[arrayIndex] == '-';
    }

    public void bookSeats(int seatNumber, char seatType) {
        int arrayIndex = seatNumber - 1;
        cinemaHall[arrayIndex] = seatType;
        noOfSeatsBooked++;
    }

    public void generateReceipt(int totalNumberOfSeatsToBook, double standardTotal,
String standardSeats,
                                double frequentTotal, String frequentSeats, double
pensionareTotal, String pensionareSeats, double personTotal,
                                int standardCount, int frequentCount, int
pensionareCount, String receiptType) {

        //Local Variables to format the output
        String lStandardTotal = String.format("%05.2f", standardTotal);
        String lFrequentTotal = String.format("%05.2f", frequentTotal);
        String lPensionerTotal = String.format("%05.2f", pensionareTotal);
        String lPersonTotal = String.format("%05.2f", personTotal);


        System.out.println();
        if (receiptType.equals("Booking")) {
            System.out.print(green);
            System.out.println("----------------Booking Receipt----------------");

        } else {
            System.out.println(red);
            System.out.println("----------------Refund Receipt----------------");
        }

        System.out.println("Total Seats: " + totalNumberOfSeatsToBook);
        System.out.println(
                standardCount + "\tx Standard Seats = $" + lStandardTotal + "\t| Seat 
Numbers: " + standardSeats);
        System.out.println(
                frequentCount + "\tx Standard Seats = $" + lFrequentTotal + "\t| Seat 
Numbers: " + frequentSeats);
        System.out.println(
                pensionareCount + "\tx Standard Seats = $" + lPensionerTotal + "\t| Seat 
Numbers: " + pensionareSeats);
        System.out.println("Total: $" + lPersonTotal);
        System.out.println("-----------------------------------------------");
        System.out.print(endColor);
    }

    public void refundSeats(int seatNumber) {
        int arrayIndex = seatNumber - 1;
        cinemaHall[arrayIndex] = '-';
        noOfSeatsBooked--;
    }

    public char returnSeatType(int seatNumber) {
        int arrayIndex = seatNumber - 1;
        return cinemaHall[arrayIndex];
    }

    public void printStatReport() {
        double avgSeatPrice = (sPrice+fPrice+pPrice)/3;
        double percentageOfSeatsSold = ((double)(noOfSeatsBooked /(double)
totalNumberOfSeats))*100;
        //Local Variables to Format the String
        String lPercentage = String.format("%05.2f", percentageOfSeatsSold);
        String lAvgPrice = String.format("%05.2f", avgSeatPrice);

        System.out.println(blue);
        System.out.println("--------------Statistics Report--------------");
        System.out.println("Date: " + date);
        System.out.println("Total Income of The Cinema: $" + totalIncome);
        System.out.println("Total Number of Seats Booked: " + noOfSeatsBooked);
        System.out.println("Average Seat Price: $"+lAvgPrice);
        System.out.println("Percentage of Seats Sold: "+lPercentage+"%");
        System.out.println("---------------------------------------------");
        System.out.println(endColor);
    }

}
```

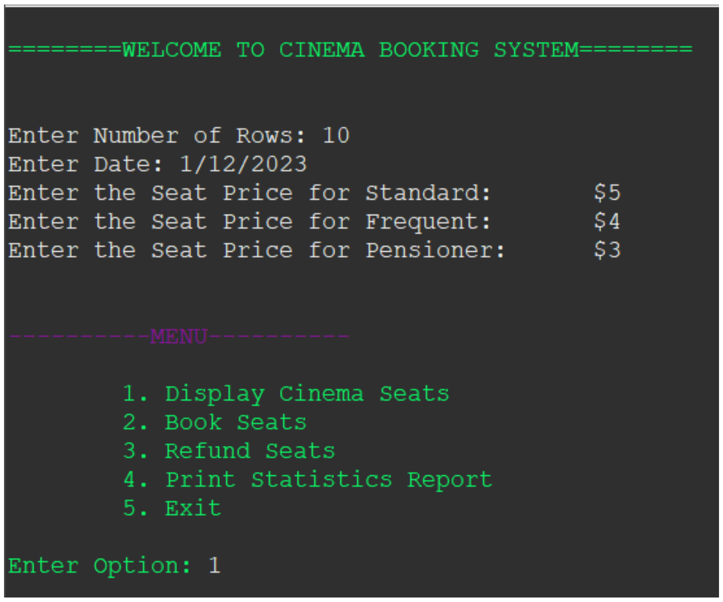
CinemaTest.java

```
package assignment1;

import static org.junit.jupiter.api.Assertions.*;

import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

class CinemaTest {
    private Cinema cinema;
    @BeforeEach
    void setUp() {
        // You can initialize the Cinema object here or in each test method
        cinema = new Cinema(5, 10, 50, "2023-01-01", 10.0, 15.0, 8.0);
        cinema.initializeCinema(); // Optional: Initialize the cinemaHall array
    }

    //JUnit test cases for isSeatAvailabe() Method

    @Test
    void atestIsSeatAvailableWhenSeatIsAvailable() {
        assertTrue(cinema.isSeatAvailable(1));
    }

    @Test
    void atestIsSeatAvailableWhenSeatIsBooked() {
        cinema.bookSeats(1, 'S');
        assertFalse(cinema.isSeatAvailable(1));
    }

    @Test
    void atestIsSeatAvailableWithInvalidSeatNumber() {
        assertThrows(IndexOutOfBoundsException.class, () -> cinema.isSeatAvailable(0));
    }


    //JUnit test cases for bookSeats() Method
    @Test
    void btestIsSeatAvailableWhenSeatIsAvailable() {
        assertTrue(cinema.isSeatAvailable(1));
    }

    @Test
    void btestIsSeatAvailableWhenSeatIsBooked() {
        cinema.bookSeats(1, 'S');
        assertFalse(cinema.isSeatAvailable(1));
    }

    @Test
    void btestIsSeatAvailableWithInvalidSeatNumber() {
        assertThrows(IndexOutOfBoundsException.class, () -> cinema.isSeatAvailable(0));
    }

    @Test
    void btestIsSeatAvailableWithInvalidNegativeSeatNumber() {
        assertThrows(IndexOutOfBoundsException.class, () -> cinema.isSeatAvailable(-1));
    }

    @Test
    void btestIsSeatAvailableWithInvalidLargeSeatNumber() {
        assertThrows(IndexOutOfBoundsException.class, () -> cinema.isSeatAvailable(60));
    }


    //JUnit test Cases for refundSeats() Method
    @Test
    void ctestRefundSeatsWhenSeatIsBooked() {
        cinema.bookSeats(1, 'S');
        cinema.refundSeats(1);
        assertTrue(cinema.isSeatAvailable(1));
        assertEquals(0, cinema.getNoOfSeatsBooked());
    }
```

```java
    @Test
    void ctestRefundSeatsWithInvalidSeatNumber() {
        assertThrows(IndexOutOfBoundsException.class, () -> cinema.refundSeats(0));
    }

    @Test
    void ctestRefundSeatsWithInvalidNegativeSeatNumber() {
        assertThrows(IndexOutOfBoundsException.class, () -> cinema.refundSeats(-1));
    }

    @Test
    void ctestRefundSeatsWithInvalidLargeSeatNumber() {
        assertThrows(IndexOutOfBoundsException.class, () -> cinema.refundSeats(60));
    }

}
```

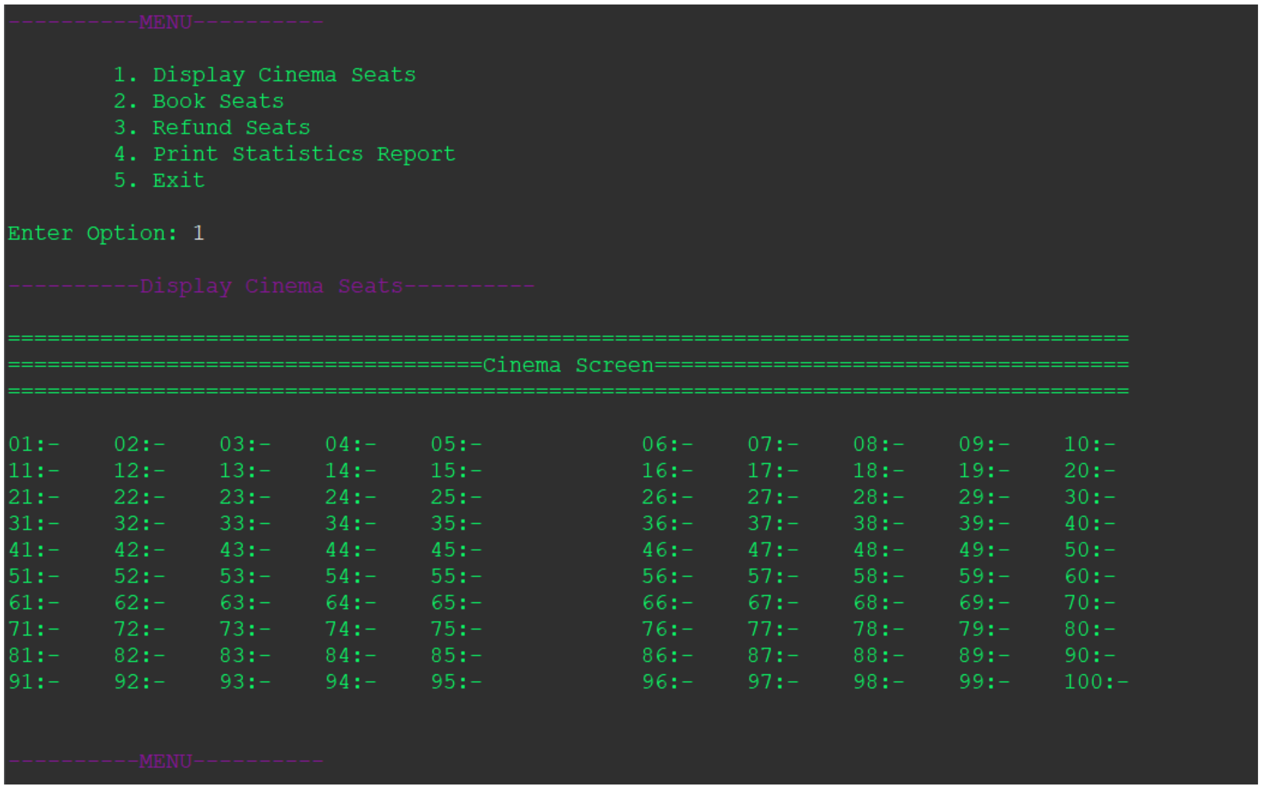
Screenshots

Initialization:



```
========WELCOME TO CINEMA BOOKING SYSTEM========


Enter Number of Rows: 10
Enter Date: 1/12/2023
Enter the Seat Price for Standard:       $5
Enter the Seat Price for Frequent:       $4
Enter the Seat Price for Pensioner:      $3


----------MENU----------

        1. Display Cinema Seats
        2. Book Seats
        3. Refund Seats
        4. Print Statistics Report
        5. Exit

Enter Option: 1
```

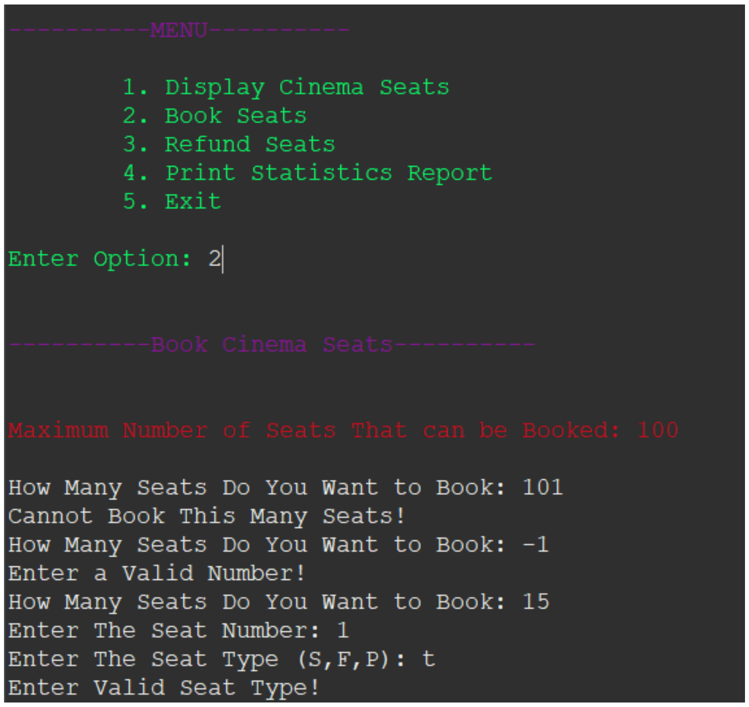
Print Cinema (With No Booked Seats):



```
----------MENU----------

        1. Display Cinema Seats
        2. Book Seats
        3. Refund Seats
        4. Print Statistics Report
        5. Exit

Enter Option: 1

----------Display Cinema Seats----------

=====================================================================================
====================================Cinema Screen====================================
=====================================================================================

01:-    02:-    03:-    04:-    05:-            06:-    07:-    08:-    09:-    10:-
11:-    12:-    13:-    14:-    15:-            16:-    17:-    18:-    19:-    20:-
21:-    22:-    23:-    24:-    25:-            26:-    27:-    28:-    29:-    30:-
31:-    32:-    33:-    34:-    35:-            36:-    37:-    38:-    39:-    40:-
41:-    42:-    43:-    44:-    45:-            46:-    47:-    48:-    49:-    50:-
51:-    52:-    53:-    54:-    55:-            56:-    57:-    58:-    59:-    60:-
61:-    62:-    63:-    64:-    65:-            66:-    67:-    68:-    69:-    70:-
71:-    72:-    73:-    74:-    75:-            76:-    77:-    78:-    79:-    80:-
81:-    82:-    83:-    84:-    85:-            86:-    87:-    88:-    89:-    90:-
91:-    92:-    93:-    94:-    95:-            96:-    97:-    98:-    99:-    100:-


----------MENU----------
```

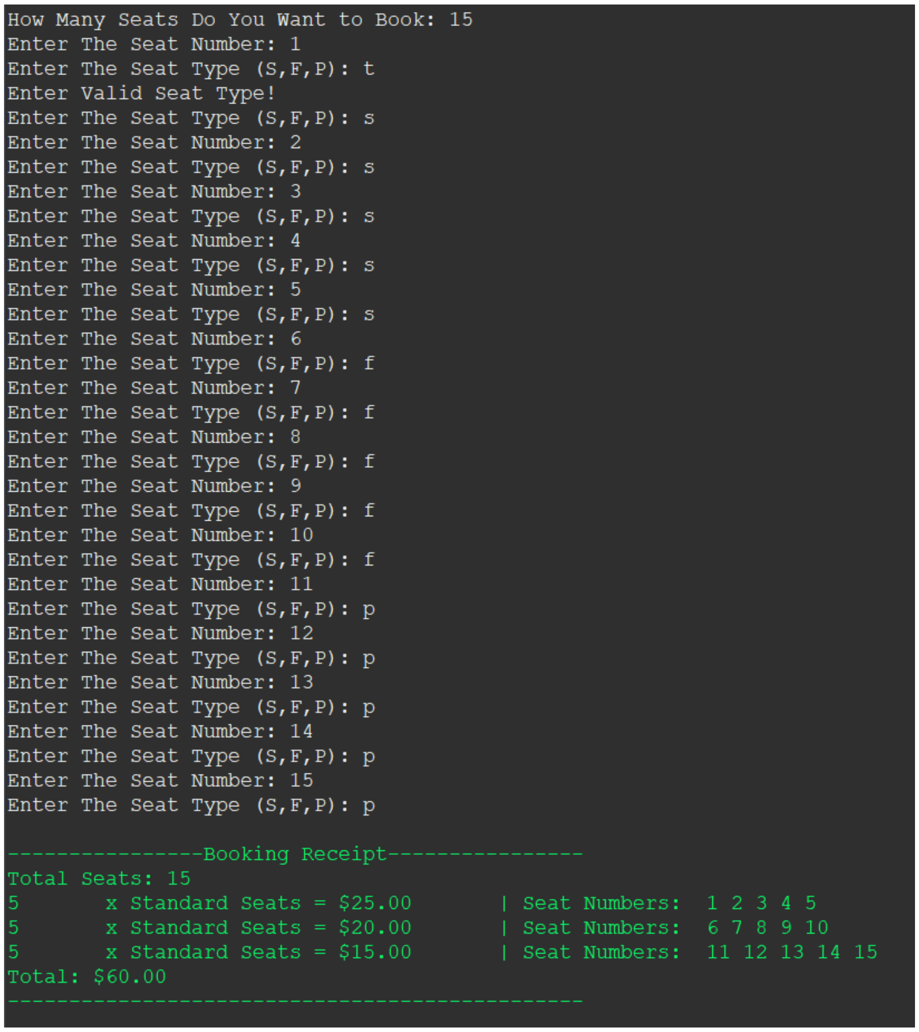
Book Seats Validations:

- Try to book More seats than Existing Seats
- Try to Book 0 or Negative Number of Seats
- Try to Book a Seat of Wrong Type (t)



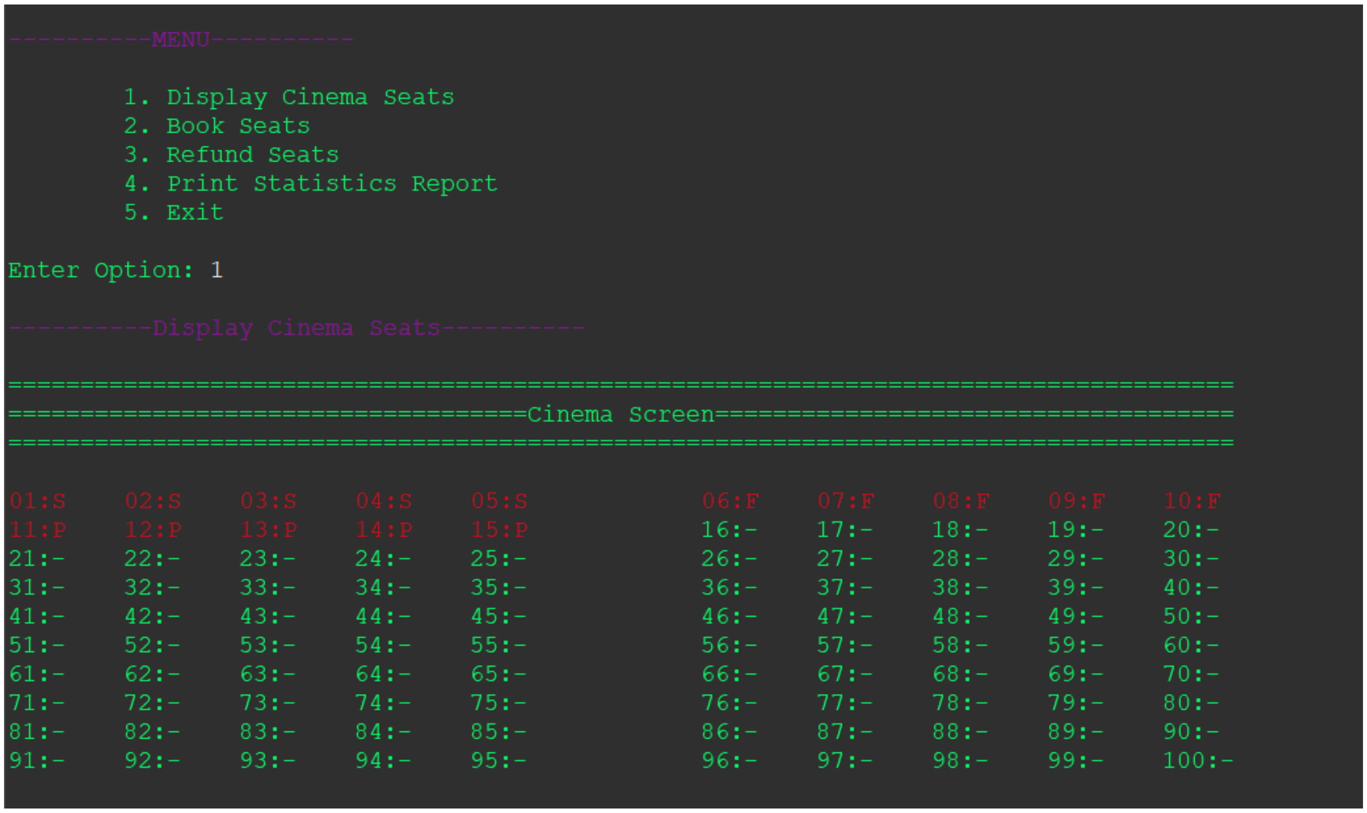
```
----------MENU----------

        1. Display Cinema Seats
        2. Book Seats
        3. Refund Seats
        4. Print Statistics Report
        5. Exit

Enter Option: 2


----------Book Cinema Seats----------


Maximum Number of Seats That can be Booked: 100

How Many Seats Do You Want to Book: 101
Cannot Book This Many Seats!
How Many Seats Do You Want to Book: -1
Enter a Valid Number!
How Many Seats Do You Want to Book: 15
Enter The Seat Number: 1
Enter The Seat Type (S,F,P): t
Enter Valid Seat Type!
```

Book Valid Range of Seats:



```
How Many Seats Do You Want to Book: 15
Enter The Seat Number: 1
Enter The Seat Type (S,F,P): t
Enter Valid Seat Type!
Enter The Seat Type (S,F,P): s
Enter The Seat Number: 2
Enter The Seat Type (S,F,P): s
Enter The Seat Number: 3
Enter The Seat Type (S,F,P): s
Enter The Seat Number: 4
Enter The Seat Type (S,F,P): s
Enter The Seat Number: 5
Enter The Seat Type (S,F,P): s
Enter The Seat Number: 6
Enter The Seat Type (S,F,P): f
Enter The Seat Number: 7
Enter The Seat Type (S,F,P): f
Enter The Seat Number: 8
Enter The Seat Type (S,F,P): f
Enter The Seat Number: 9
Enter The Seat Type (S,F,P): f
Enter The Seat Number: 10
Enter The Seat Type (S,F,P): f
Enter The Seat Number: 11
Enter The Seat Type (S,F,P): p
Enter The Seat Number: 12
Enter The Seat Type (S,F,P): p
Enter The Seat Number: 13
Enter The Seat Type (S,F,P): p
Enter The Seat Number: 14
Enter The Seat Type (S,F,P): p
Enter The Seat Number: 15
Enter The Seat Type (S,F,P): p

----------------Booking Receipt----------------
Total Seats: 15
5       x Standard Seats = $25.00       | Seat Numbers:  1 2 3 4 5
5       x Standard Seats = $20.00       | Seat Numbers:  6 7 8 9 10
5       x Standard Seats = $15.00       | Seat Numbers:  11 12 13 14 15
Total: $60.00
-----------------------------------------------
```

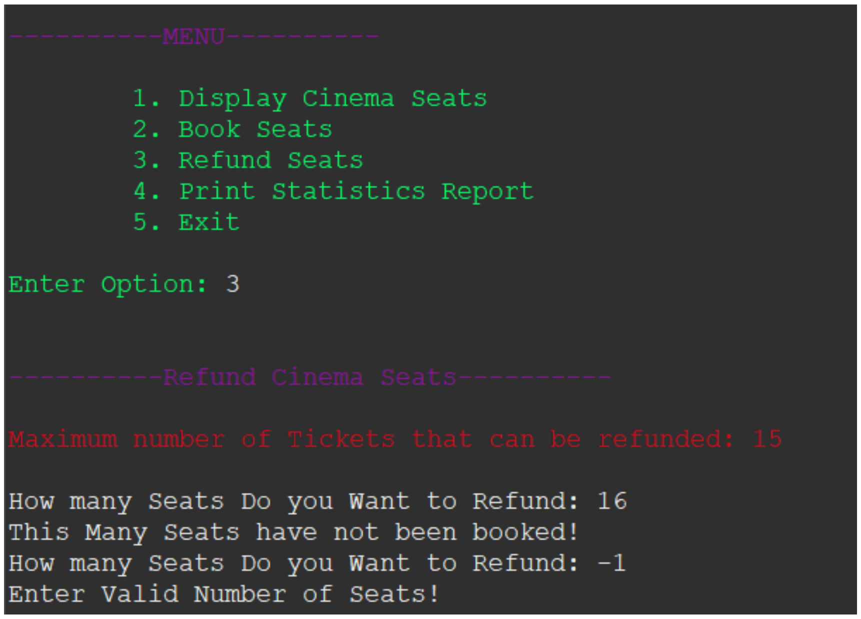
Display Seats (With Booked Seats)



```
----------MENU----------

        1. Display Cinema Seats
        2. Book Seats
        3. Refund Seats
        4. Print Statistics Report
        5. Exit

Enter Option: 1

----------Display Cinema Seats----------

=====================================================================================
====================================Cinema Screen====================================
=====================================================================================

01:S    02:S    03:S    04:S    05:S            06:F    07:F    08:F    09:F    10:F
11:P    12:P    13:P    14:P    15:P            16:-    17:-    18:-    19:-    20:-
21:-    22:-    23:-    24:-    25:-            26:-    27:-    28:-    29:-    30:-
31:-    32:-    33:-    34:-    35:-            36:-    37:-    38:-    39:-    40:-
41:-    42:-    43:-    44:-    45:-            46:-    47:-    48:-    49:-    50:-
51:-    52:-    53:-    54:-    55:-            56:-    57:-    58:-    59:-    60:-
61:-    62:-    63:-    64:-    65:-            66:-    67:-    68:-    69:-    70:-
71:-    72:-    73:-    74:-    75:-            76:-    77:-    78:-    79:-    80:-
81:-    82:-    83:-    84:-    85:-            86:-    87:-    88:-    89:-    90:-
91:-    92:-    93:-    94:-    95:-            96:-    97:-    98:-    99:-    100:-
```

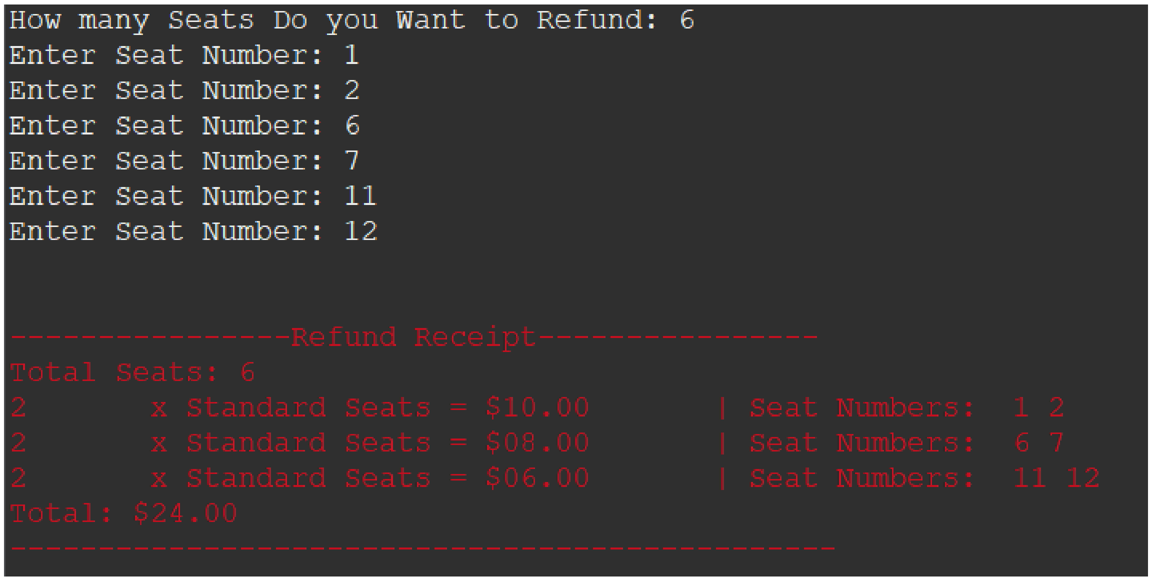
Refund Validations:

- Refund Seats More than the Booked amount
- Refund negative Number of Seats



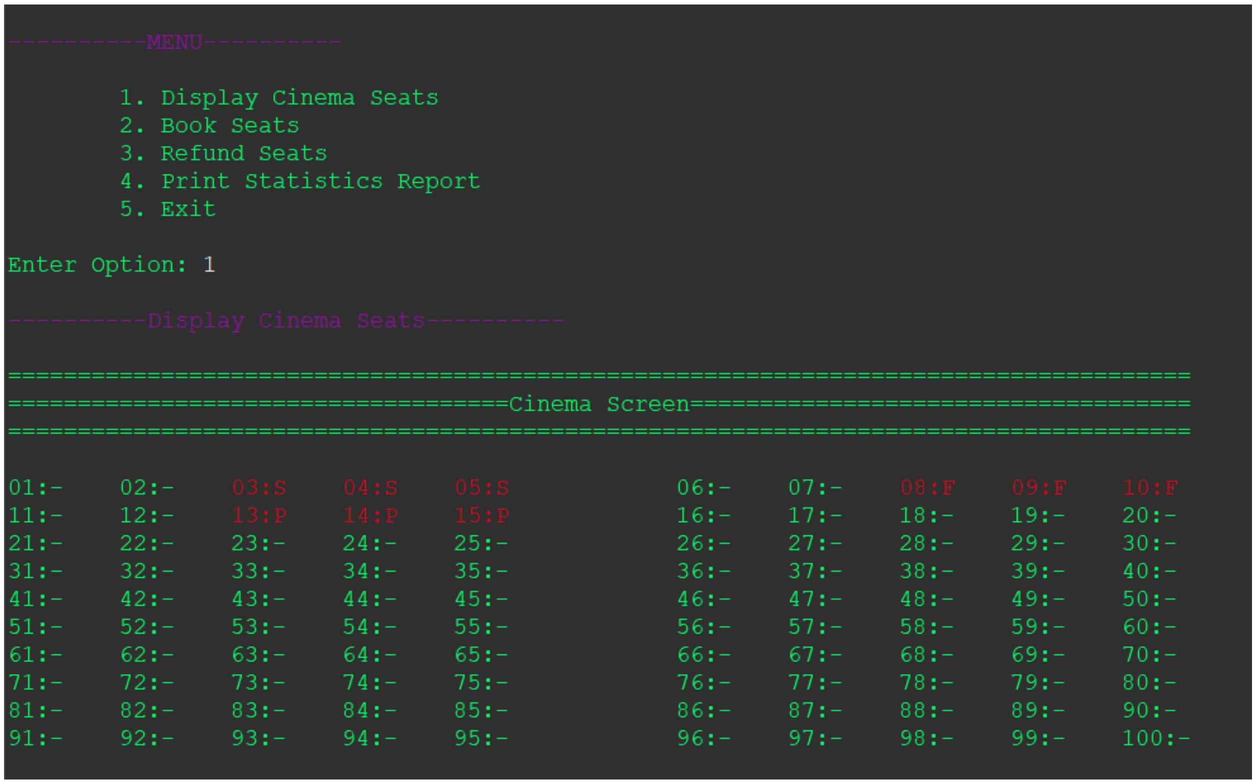
```
----------MENU----------

        1. Display Cinema Seats
        2. Book Seats
        3. Refund Seats
        4. Print Statistics Report
        5. Exit

Enter Option: 3


----------Refund Cinema Seats----------

Maximum number of Tickets that can be refunded: 15

How many Seats Do you Want to Refund: 16
This Many Seats have not been booked!
How many Seats Do you Want to Refund: -1
Enter Valid Number of Seats!
```

Refund Valid Number of Seats:



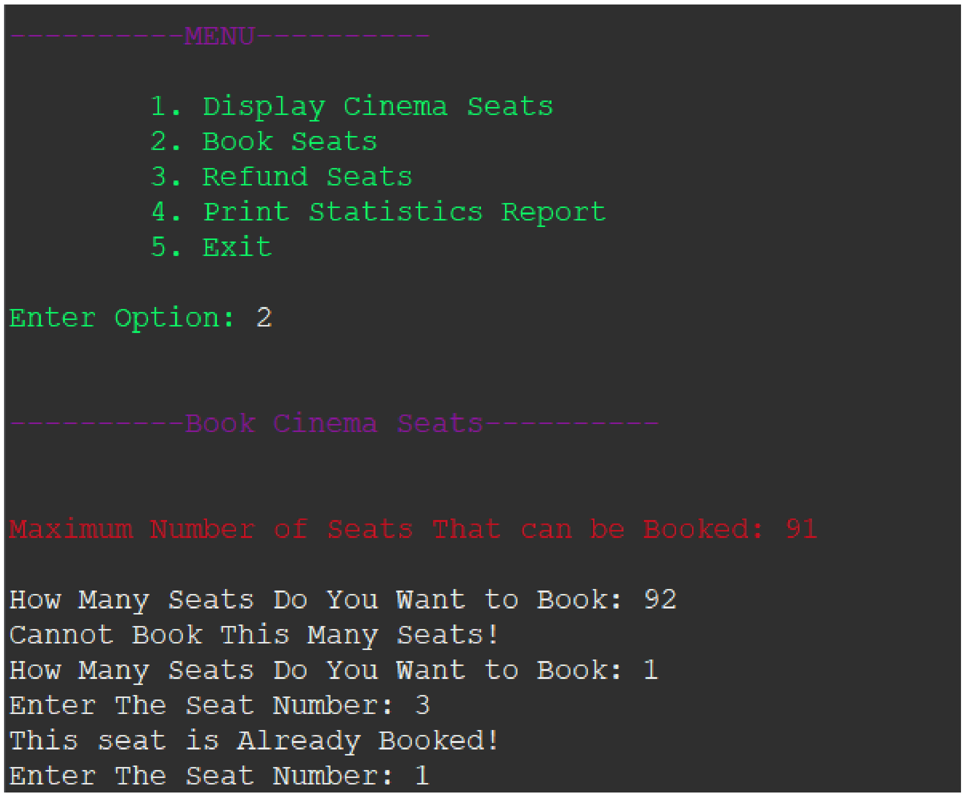
```
How many Seats Do you Want to Refund: 6
Enter Seat Number: 1
Enter Seat Number: 2
Enter Seat Number: 6
Enter Seat Number: 7
Enter Seat Number: 11
Enter Seat Number: 12


----------------Refund Receipt----------------
Total Seats: 6
2       x Standard Seats = $10.00       | Seat Numbers:  1 2
2       x Standard Seats = $08.00       | Seat Numbers:  6 7
2       x Standard Seats = $06.00       | Seat Numbers:  11 12
Total: $24.00
-----------------------------------------------
```

Display Seats after Refund:



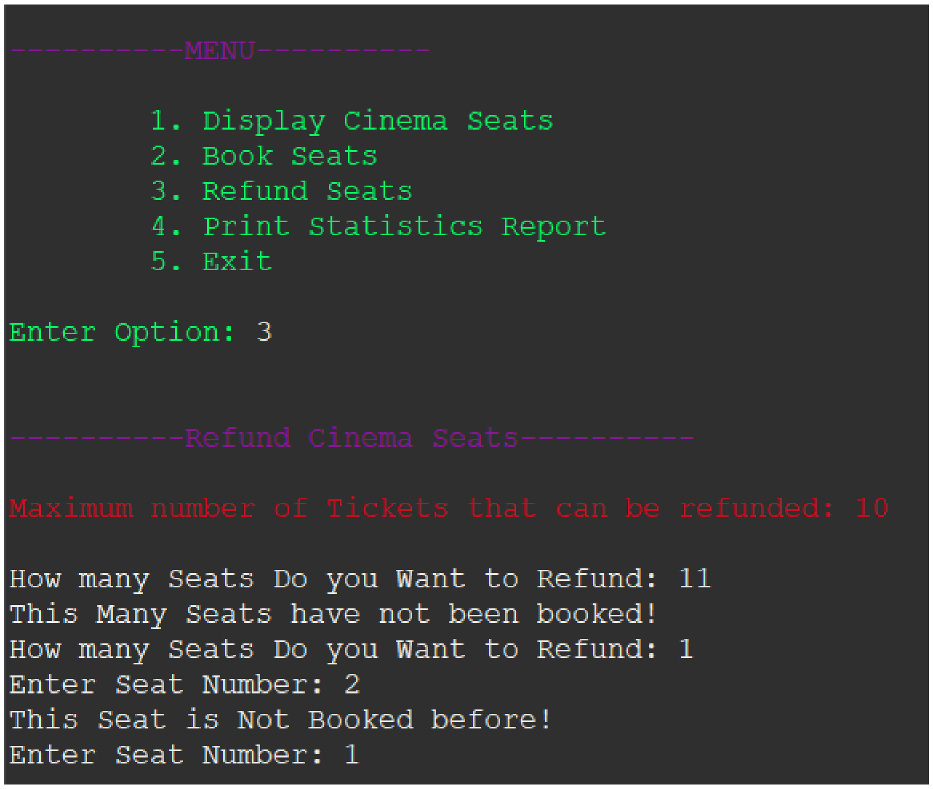
```
----------MENU----------

        1. Display Cinema Seats
        2. Book Seats
        3. Refund Seats
        4. Print Statistics Report
        5. Exit

Enter Option: 1

----------Display Cinema Seats----------

====================================================================================
====================================Cinema Screen===================================
====================================================================================

01:-    02:-    03:S    04:S    05:S            06:-    07:-    08:F    09:F    10:F
11:-    12:-    13:P    14:P    15:P            16:-    17:-    18:-    19:-    20:-
21:-    22:-    23:-    24:-    25:-            26:-    27:-    28:-    29:-    30:-
31:-    32:-    33:-    34:-    35:-            36:-    37:-    38:-    39:-    40:-
41:-    42:-    43:-    44:-    45:-            46:-    47:-    48:-    49:-    50:-
51:-    52:-    53:-    54:-    55:-            56:-    57:-    58:-    59:-    60:-
61:-    62:-    63:-    64:-    65:-            66:-    67:-    68:-    69:-    70:-
71:-    72:-    73:-    74:-    75:-            76:-    77:-    78:-    79:-    80:-
81:-    82:-    83:-    84:-    85:-            86:-    87:-    88:-    89:-    90:-
91:-    92:-    93:-    94:-    95:-            96:-    97:-    98:-    99:-    100:-
```

Attempt to Book a seat that is already booked:



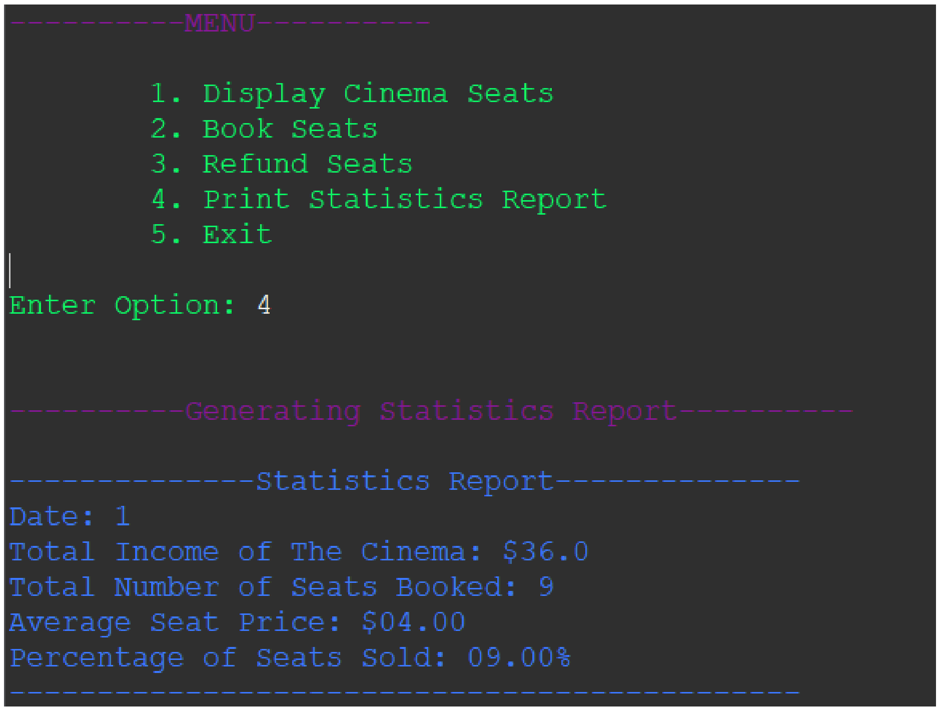
```
----------MENU----------

        1. Display Cinema Seats
        2. Book Seats
        3. Refund Seats
        4. Print Statistics Report
        5. Exit

Enter Option: 2


----------Book Cinema Seats----------


Maximum Number of Seats That can be Booked: 91

How Many Seats Do You Want to Book: 92
Cannot Book This Many Seats!
How Many Seats Do You Want to Book: 1
Enter The Seat Number: 3
This seat is Already Booked!
Enter The Seat Number: 1
```

Refund a ticket that is not booked before:



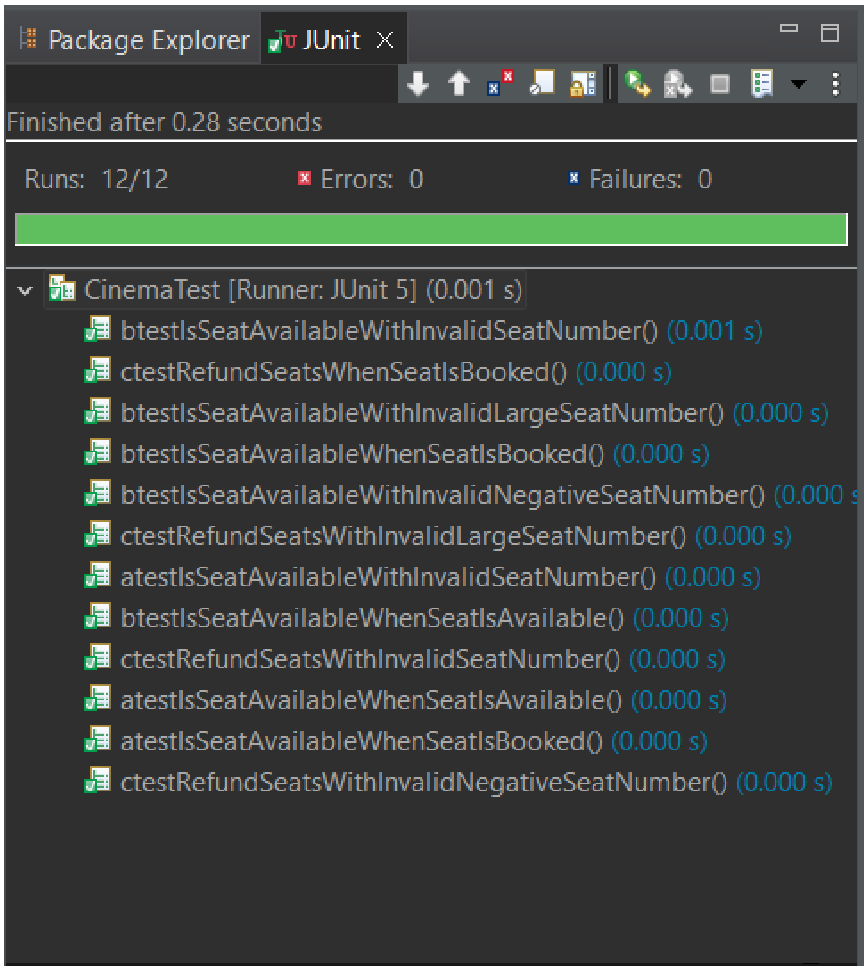
```
----------MENU----------

        1. Display Cinema Seats
        2. Book Seats
        3. Refund Seats
        4. Print Statistics Report
        5. Exit

Enter Option: 3


----------Refund Cinema Seats----------

Maximum number of Tickets that can be refunded: 10

How many Seats Do you Want to Refund: 11
This Many Seats have not been booked!
How many Seats Do you Want to Refund: 1
Enter Seat Number: 2
This Seat is Not Booked before!
Enter Seat Number: 1
```

Print Statistics Report:



```
----------MENU----------

        1. Display Cinema Seats
        2. Book Seats
        3. Refund Seats
        4. Print Statistics Report
        5. Exit

Enter Option: 4


----------Generating Statistics Report----------

--------------Statistics Report--------------
Date: 1
Total Income of The Cinema: $36.0
Total Number of Seats Booked: 9
Average Seat Price: $04.00
Percentage of Seats Sold: 09.00%
---------------------------------------------
```

JUnit Tests:



```
Finished after 0.28 seconds

Runs: 12/12          Errors: 0          Failures: 0

CinemaTest [Runner: JUnit 5] (0.001 s)
    btestIsSeatAvailableWithInvalidSeatNumber() (0.001 s)
    ctestRefundSeatsWhenSeatIsBooked() (0.000 s)
    btestIsSeatAvailableWithInvalidLargeSeatNumber() (0.000 s)
    btestIsSeatAvailableWhenSeatIsBooked() (0.000 s)
    btestIsSeatAvailableWithInvalidNegativeSeatNumber() (0.000 s
    ctestRefundSeatsWithInvalidLargeSeatNumber() (0.000 s)
    atestIsSeatAvailableWithInvalidSeatNumber() (0.000 s)
    btestIsSeatAvailableWhenSeatIsAvailable() (0.000 s)
    ctestRefundSeatsWithInvalidSeatNumber() (0.000 s)
    atestIsSeatAvailableWhenSeatIsAvailable() (0.000 s)
    atestIsSeatAvailableWhenSeatIsBooked() (0.000 s)
    ctestRefundSeatsWithInvalidNegativeSeatNumber() (0.000 s)
```