MALNAD COLLEGE OF ENGINEERING

(An Autonomous Institution under Visvesvaraya Technological University, Belagavi)

Hassan – 573202, Karnataka, India

Course title: DATA STRUCTURES

Course code: (23AI304)

Project based Learning:

Design a flight seat allocation system using arrays.

Submitted by:

MADHURYA.H.A.	4MC23CI027
YUKTHA.N.P.	4MC23CI062
YUVARAJ	4MC23CI063

Under the guidance of:

DR BALAJI PRABHU B V

Associate Professor and HOD

Dept. of CSE (AI&ML)

Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning)
2024-2025

Problem statement:

Design a flight seat allocation system using arrays. Your system should allow passengers to book seats, cancel bookings, and view the current seating arrangements. Include features to display available seats and handle invalid inputs gracefully.

This project is a Flight Seat Allocation System implemented in C. The purpose of the project is to manage seat allocation for a flight, including viewing, booking, cancelling, and displaying available seats. Below is a detailed step-by-step explanation:

1. Pre-processor and Constants

```
#include <stdio.h>

#define ROWS 5

#define COLUMNS 4
```

The program includes the standard I/O library for input and output operations.

ROWS and COLUMNS are defined constants to represent the number of rows (5) and columns (4) in the seating arrangement.

2. Functions

The program uses modular functions to handle specific tasks:

a) displaySeats

This function displays the current seating arrangement.

1. Define the function and accept a 2D array.
2. Loop over rows.
3. For each row, loop over columns.
4. Print each seat value.
5. After each row, print a new line.

This results in a formatted display of the seating arrangement row by row.

b) displayAvailableSeats

This function identifies and lists all the seats marked 'O' (available).

It counts the number of available seats and displays their respective row and column numbers.

If no seats are available, it displays a message saying “No seats available!”.

c) bookSeat

Input Validation: Ensures the entered row and column are within the bounds of the seating arrangement.

If the seat is already marked 'X', it informs the user that the seat is taken.

Otherwise, it marks the seat as 'X' (booked) and confirms the booking.

d) cancelBooking

Input Validation: Checks the entered row and column values.

If the seat is already 'O' (available), it indicates there's no booking to cancel.

Otherwise, it marks the seat as 'O' (available) and confirms the cancellation.

3. main Function

a) Initialization

A 2D character array (seats) is initialized to represent the seating arrangement.

All seats are initially set to 'O' (available).

b) Menu-driven System

A loop is used to repeatedly display the menu and prompt the user for a choice.

The user selects from 5 options:

1. View seating arrangement.

2. Display available seats.

3. Book a seat.

4. Cancel a booking.

5. Exit the program.

c) Switch-case for Menu Handling

Depending on the user's choice, the appropriate function is called.

If the user enters an invalid option, a message is displayed.

Choosing option 5 exits the loop and ends the program.

4. Execution Flow

1. Start: The program begins by initializing all seats as available ('O').

2. Menu Display: The user is shown a menu and prompted to select an option.

3. User Interaction:

If the user chooses:

Option 1: The current seating arrangement is displayed.

Option 2: All available seats are listed.

Option 3: The user can book a seat, with input validation for proper selection.

Option 4: The user can cancel a booking, with input validation.

Option 5: The program exits.

4. Repeat:

After each operation, the menu is displayed again until the user exits.

5. Key Concepts:

2D Arrays: Used to represent the seating arrangement.

Input Validation: Ensures valid rows and columns are selected.

Switch-Case: Implements the menu system.

Infinite Loop: Ensures the program runs continuously until the user exits.

This program effectively demonstrates modular programming, input validation, and menu-driven design for a simple seat allocation system.

OUTPUT:

The program implements a flight seat allocation system using arrays, allowing users to book seats, cancel bookings, view the current seating arrangement, and display available seats. Here's a detailed explanation:

1. Objective

The system uses a 2D array to represent the seating arrangement. Each seat is represented by either 'O' (available) or 'X' (booked). The program provides options for the user to interact with the seating arrangement, fulfilling all the requirements outlined in the problem.

2. Program Structure

The program is divided into the following components:

a. Array Initialization

A 2D array seats [ROWS][COLUMNS] represents the seating arrangement.

'O' denotes an available seat, and 'X' denotes a booked seat.

The seating is initialized with all seats marked as 'O'.

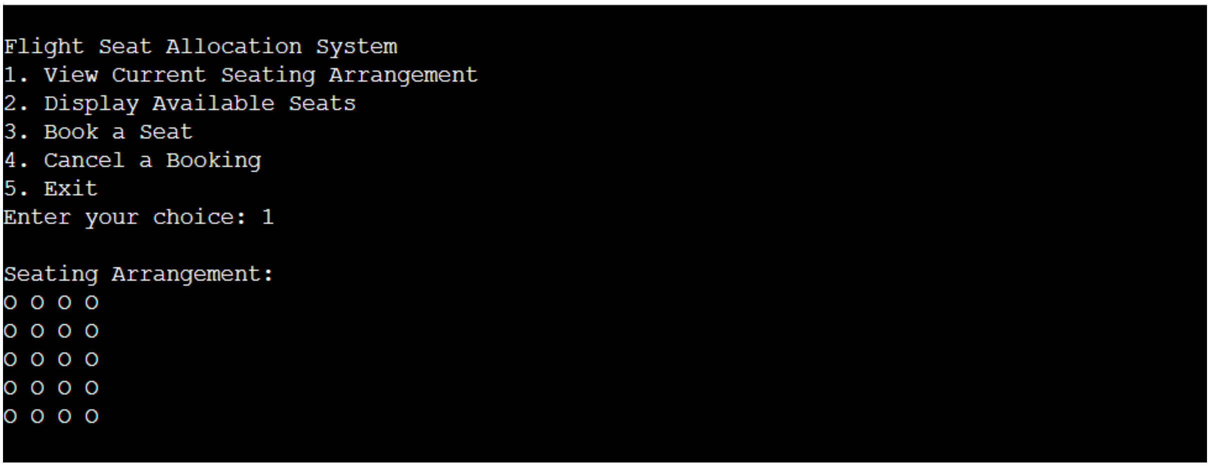
b. Menu-Driven Interface

The user is presented with a menu for different operations.

The do-while loop ensures the menu is displayed repeatedly until the user chooses to exit (choice == 5).

3. Functionalities

a. Display Seating Arrangement



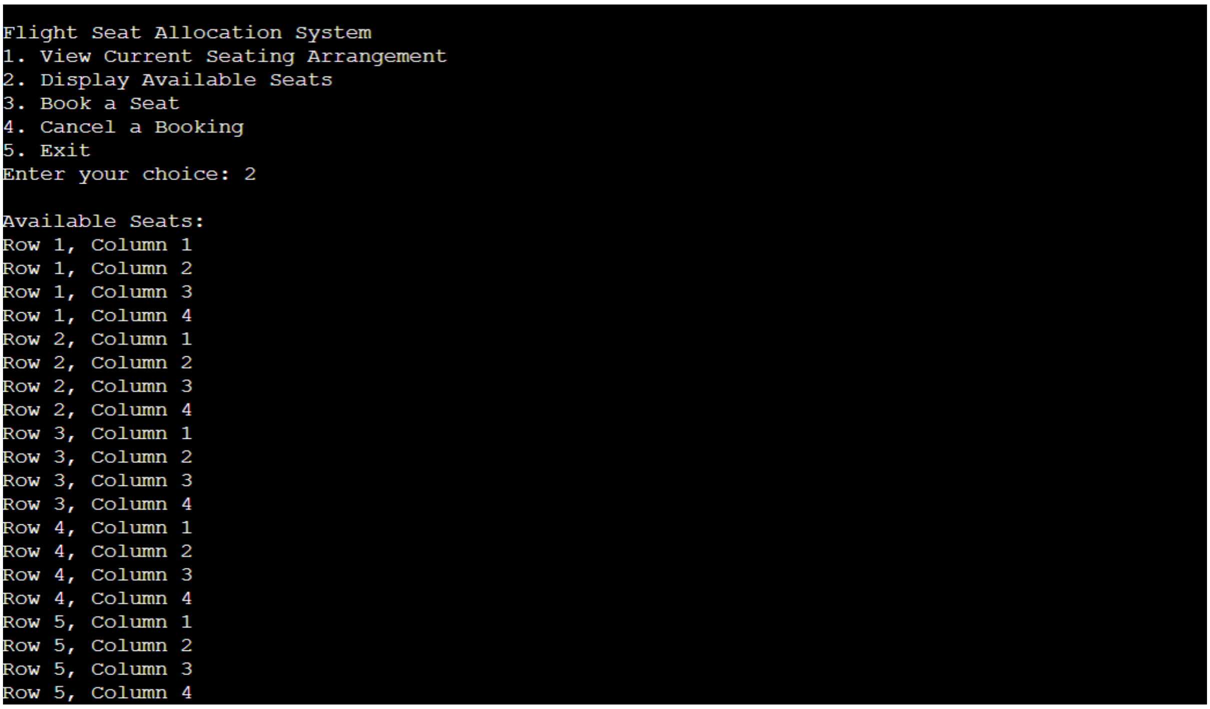
```
Flight Seat Allocation System
1. View Current Seating Arrangement
2. Display Available Seats
3. Book a Seat
4. Cancel a Booking
5. Exit
Enter your choice: 1

Seating Arrangement:
O O O O
O O O O
O O O O
O O O O
O O O O
```

Displays the current status of each seat (booked or available).

Seats are printed in a grid format for clarity.

b. Display Available Seats



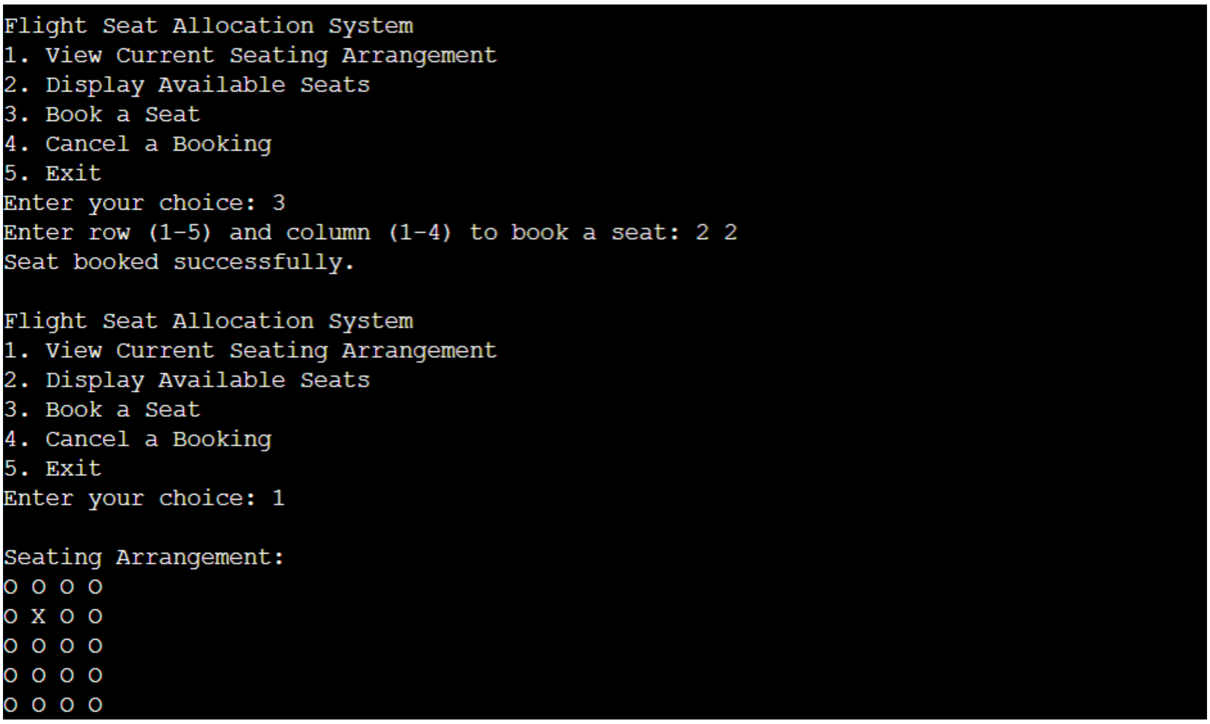
```
Flight Seat Allocation System
1. View Current Seating Arrangement
2. Display Available Seats
3. Book a Seat
4. Cancel a Booking
5. Exit
Enter your choice: 2

Available Seats:
Row 1, Column 1
Row 1, Column 2
Row 1, Column 3
Row 1, Column 4
Row 2, Column 1
Row 2, Column 2
Row 2, Column 3
Row 2, Column 4
Row 3, Column 1
Row 3, Column 2
Row 3, Column 3
Row 3, Column 4
Row 4, Column 1
Row 4, Column 2
Row 4, Column 3
Row 4, Column 4
Row 5, Column 1
Row 5, Column 2
Row 5, Column 3
Row 5, Column 4
```

Checks the array for all available seats ('O') and displays their row and column positions.

If no seats are available, it informs the user.

c. Book a Seat



```
Flight Seat Allocation System
1. View Current Seating Arrangement
2. Display Available Seats
3. Book a Seat
4. Cancel a Booking
5. Exit
Enter your choice: 3
Enter row (1-5) and column (1-4) to book a seat: 2 2
Seat booked successfully.

Flight Seat Allocation System
1. View Current Seating Arrangement
2. Display Available Seats
3. Book a Seat
4. Cancel a Booking
5. Exit
Enter your choice: 1

Seating Arrangement:
O O O O
O X O O
O O O O
O O O O
O O O O
```

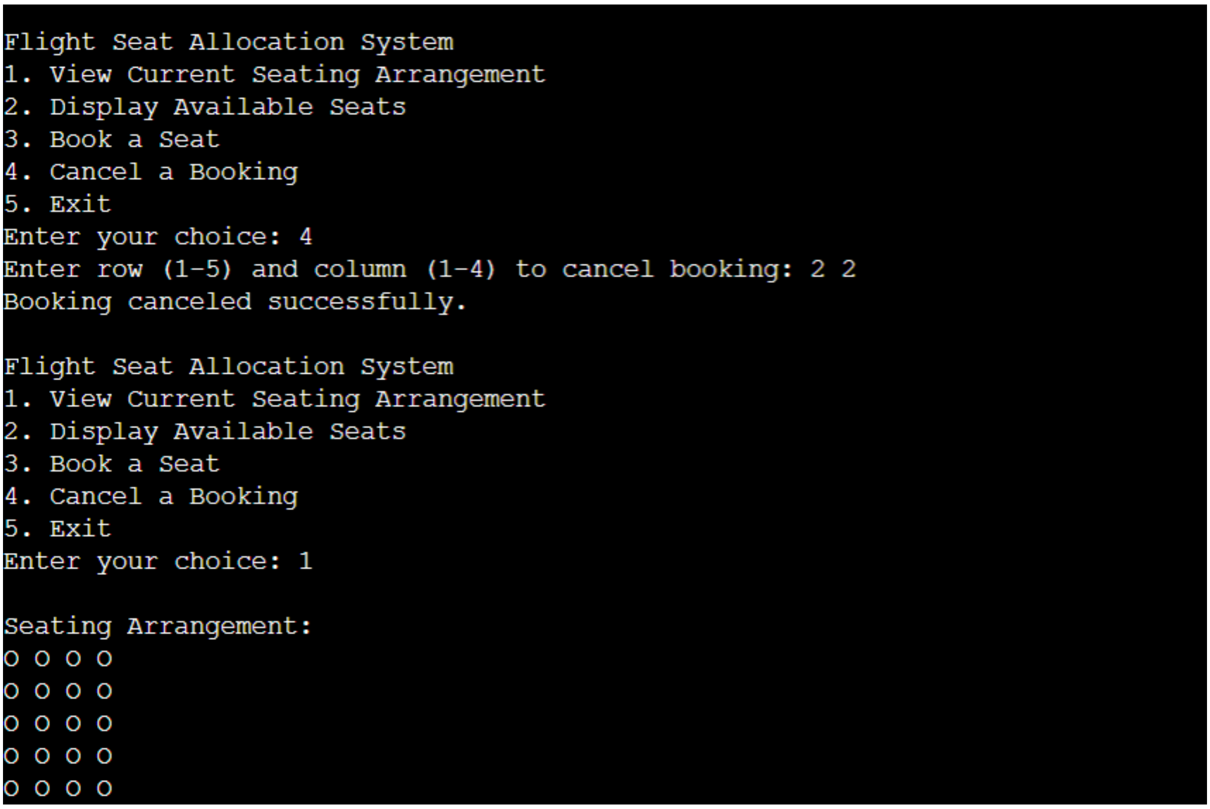
Prompts the user for a row and column to book a seat.

Validates the input to ensure it falls within valid rows and columns.

If the seat is available ('O'), it is booked ('X').

If the seat is already booked, the user is notified.

d. Cancel a Booking



```
Flight Seat Allocation System
1. View Current Seating Arrangement
2. Display Available Seats
3. Book a Seat
4. Cancel a Booking
5. Exit
Enter your choice: 4
Enter row (1-5) and column (1-4) to cancel booking: 2 2
Booking canceled successfully.

Flight Seat Allocation System
1. View Current Seating Arrangement
2. Display Available Seats
3. Book a Seat
4. Cancel a Booking
5. Exit
Enter your choice: 1

Seating Arrangement:
O O O O
O O O O
O O O O
O O O O
O O O O
```

Prompts the user for a row and column to cancel a booking.

Validates the input to ensure it falls within valid rows and columns.

If the seat is booked ('X'), the booking is cancelled ('O').

If the seat is already available, the user is notified.

4. Input Validation

The program ensures:

Rows and columns are within valid bounds:

if (row < 1 || row > ROWS || col < 1 || col > COLUMNS)

Prevents double booking or cancellation of non-booked seats.

5. Example Execution

1. User views the seating arrangement:

O O O O

O O O O

O O O O

O O O O

O O O O

2. User books a seat at Row 2, Column 3:

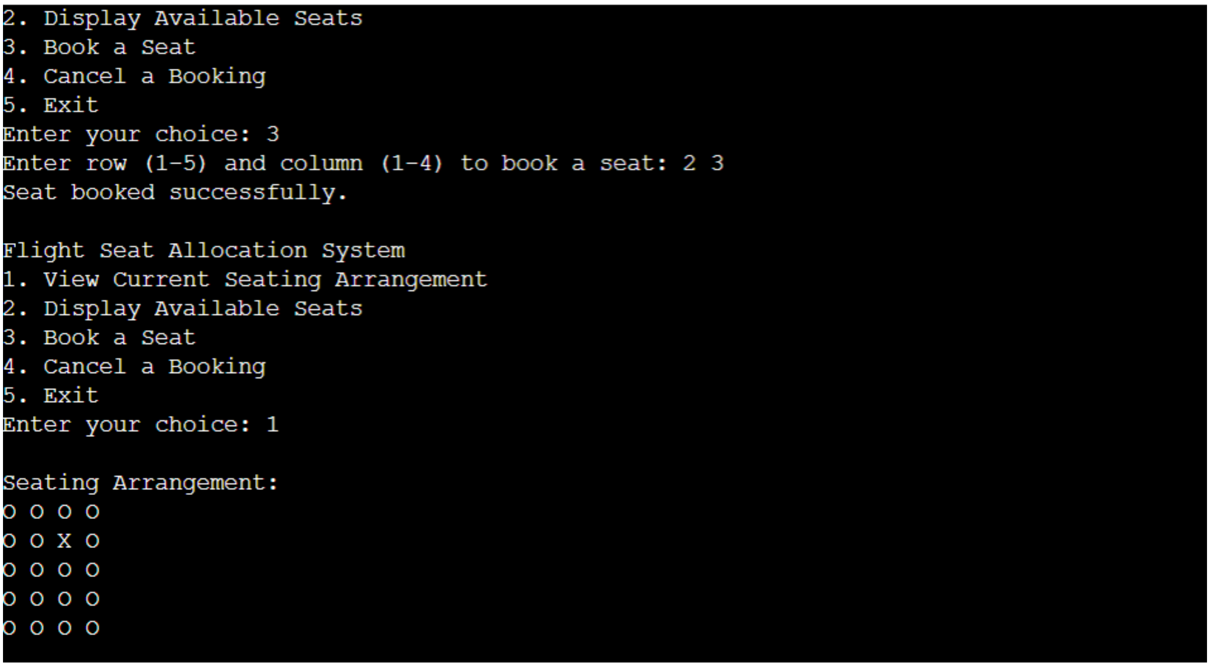
O O O O

O O X O

O O O O

O O O O

O O O O



```
2. Display Available Seats
3. Book a Seat
4. Cancel a Booking
5. Exit
Enter your choice: 3
Enter row (1-5) and column (1-4) to book a seat: 2 3
Seat booked successfully.

Flight Seat Allocation System
1. View Current Seating Arrangement
2. Display Available Seats
3. Book a Seat
4. Cancel a Booking
5. Exit
Enter your choice: 1

Seating Arrangement:
O O O O
O O X O
O O O O
O O O O
O O O O
```

3. User cancels the booking at Row 2, Column 3:

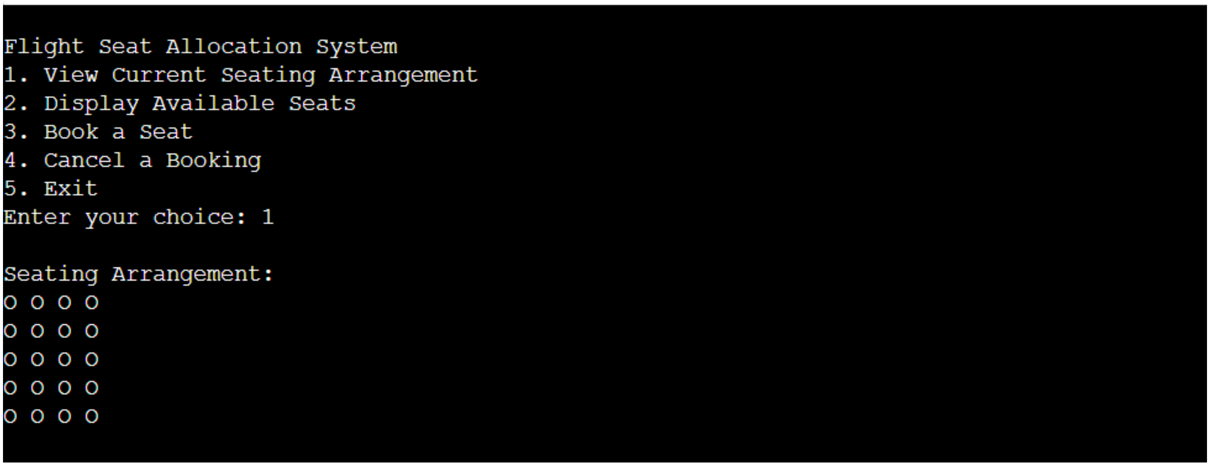
O O O O

O O O O

O O O O

O O O O

O O O O



```
Flight Seat Allocation System
1. View Current Seating Arrangement
2. Display Available Seats
3. Book a Seat
4. Cancel a Booking
5. Exit
Enter your choice: 1

Seating Arrangement:
O O O O
O O O O
O O O O
O O O O
O O O O
```

6. Features

View Seats: Shows the complete seating arrangement.

Book a Seat: Allows booking of available seats.

Cancel a Booking: Cancels a previously booked seat.

Display Available Seats: Lists all available seats by row and column.

7. Conclusion

This program is efficient, user-friendly, and meets all requirements:

Handles invalid inputs gracefully.

Displays current and available seats.

Allows seamless booking and cancellation.

The modular design makes it easy to understand, and the menu-driven approach ensures simplicity for users.