Experiment No. 5

Aim:

To implement a ticket booking system for Cinemax Theatre using a doubly linked list for efficient management of available and booked seats.

Objective:

 To represent seat allocation in a theatre using linked data structures.

 To maintain a list of free and booked seats dynamically.

 To allow operations such as:

- Displaying available seats
- Booking seats
- Cancelling bookings

 To understand the practical application of **doubly linked lists** in real-life scenarios.

Problem Statement:

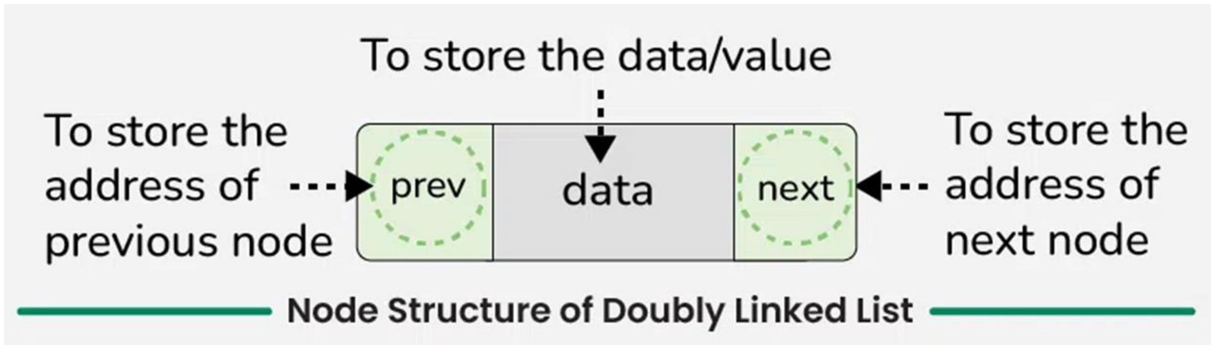
Design and implement a ticket booking system for a theatre having 10 seats. A doubly linked list should be maintained to keep track of free seats. The system should start with some randomly booked seats. On demand, the following operations should be performed:

- Display the list of available seats.
- Book the required seat(s).
- Cancel a booking.

Theory:

In computer science, a doubly linked list (DLL) is a dynamic data structure consisting of nodes where each node contains:

- **Data field** → stores seat information (seat number, status).
- **Pointer to the next node** → links to the next seat in the row.
- **Pointer to the previous node** → links to the previous seat in the row.


To store the data/value
To store the address of previous node
prev
data
next
To store the address of next node
Node Structure of Doubly Linked List

Why DLL is suitable here?

- Efficient insertion and deletion (needed for booking and cancellation).
- Easy traversal in both directions to check available seats.
- Flexibility to update seat availability dynamically.

Thus, DLL provides a realistic model for managing theatre seat booking systems where frequent updates occur.

Algorithm:

1. **Start**
2. Initialize a doubly linked list of size 10, each node representing a seat with
    - Seat number
    - Availability status (0 = free, 1 = booked)
3. Randomly assign some seats as booked initially.
4. Provide a menu-driven system:

Option 1: Display Available Seats

a. Traverse the list and print all free seat numbers.

Option 2: Book a Seat

b. Search for the given seat number.
c. If available → mark it as booked.
d. Else → display "Seat already booked."

Option 3: Cancel a Booking

e. Search for the given seat number.

f. If booked → mark it as free.
g. Else → display "Seat not booked yet."

Option 4: Exit

5. Repeat until the user exits.
6. **Stop**

Input:

Number of seats: 10

Randomly booked seats at the start

User menu choice:

1. Display available seats
2. Book seat number `n`
3. Cancel booking for seat number `n`

Output:

Note: Write output of your program

Conclusion:

The theatre ticket booking system was successfully implemented using a doubly linked list. It efficiently manages the seat allocation, provides user-friendly operations, and demonstrates the real-world application of dynamic data structures.