```cpp
#include <iostream>
#include <iomanip>
using namespace std;

struct Seat {
    int seatNumber;
    bool isBooked;
    Seat* next;
    Seat* prev;

    Seat(int num) : seatNumber(num), isBooked(false), next(nullptr), prev(nullptr) {}
};

class Row {
public:
    Seat* head;

    Row() : head(nullptr) {}

    void createRow(int seats) {
        for (int i = 1; i <= seats; ++i) {
            Seat* newSeat = new Seat(i);
            if (!head) {
                head = newSeat;
                head->next = head;
                head->prev = head;
            }
            else {
                Seat* tail = head->prev;
                tail->next = newSeat;
                newSeat->prev = tail;
                newSeat->next = head;
                head->prev = newSeat;
            }
        }
    }

    void displayAvailableSeats() const {
        Seat* temp = head;
        bool hasAvailable = false;
        do {
            if (!temp->isBooked) {
                cout << "Seat " << temp->seatNumber << " ";
                hasAvailable = true;
        }
        temp = temp->next;
    } while (temp != head);

    if (!hasAvailable) {
        cout << "All seats are booked.";
    }
    cout << endl;
}

bool bookSeat(int seatNum) {
    Seat* temp = head;
    do {
        if (temp->seatNumber == seatNum) {
            if (!temp->isBooked) {
                temp->isBooked = true;
                cout << "Seat " << seatNum << " booked successfully.\n";
                return true;
            } else {
                cout << "Seat " << seatNum << " is already booked.\n";
                return false;
            }
        }
        temp = temp->next;
    } while (temp != head);
    cout << "Seat " << seatNum << " not found.\n";
    return false;
}

bool cancelBooking(int seatNum) {
    Seat* temp = head;
    do {
        if (temp->seatNumber == seatNum) {
            if (temp->isBooked) {
                temp->isBooked = false;
                cout << "Seat " << seatNum << " booking cancelled successfully.\n";
                return true;
            } else {
                cout << "Seat " << seatNum << " is not booked.\n";
                return false;
            }
        }
        temp = temp->next;
    } while (temp != head);
```

```
        cout << "Seat " << seatNum << " not found.\n";
        return false;
    }
};

class Theater {
private:
    Row rows[10];

public:
    Theater() {
        for (int i = 0; i < 10; ++i) {
            rows[i].createRow(7);
        }
        // Initialize some random bookings for demonstration
        rows[0].bookSeat(1);
        rows[0].bookSeat(3);
        rows[1].bookSeat(5);
        rows[2].bookSeat(2);
        rows[3].bookSeat(7);
    }

    void displayAvailableSeats() const {
        for (int i = 0; i < 10; ++i) {
            cout << "Row " << i + 1 << ": ";
            rows[i].displayAvailableSeats();
        }
    }

    void bookSeat(int row, int seatNum) {
        if (row < 1 || row > 10) {
            cout << "Invalid row number.\n";
            return;
        }
        rows[row - 1].bookSeat(seatNum);
    }

    void cancelBooking(int row, int seatNum) {
        if (row < 1 || row > 10) {
            cout << "Invalid row number.\n";
            return;
        }
        rows[row - 1].cancelBooking(seatNum);
    }
```

```
};

int main() {
    Theater cinemax;
    int choice, row, seatNum;

    while (true) {
        cout << "\n--- Cinemax Ticket Booking System ---\n";
        cout << "1. Display Available Seats\n";
        cout << "2. Book a Seat\n";
        cout << "3. Cancel Booking\n";
        cout << "4. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice) {
            case 1:
                cinemax.displayAvailableSeats();
                break;
            case 2:
                cout << "Enter Row (1-10) and Seat Number (1-7) to book: ";
                cin >> row >> seatNum;
                cinemax.bookSeat(row, seatNum);
                break;
            case 3:
                cout << "Enter Row (1-10) and Seat Number (1-7) to cancel: ";
                cin >> row >> seatNum;
                cinemax.cancelBooking(row, seatNum);
                break;
            case 4:
                cout << "Exiting...\n";
                return 0;
            default:
                cout << "Invalid choice. Try again.\n";
        }
    }
}
```