SRI VENKATESWARA COLLEGE OF ENGINEERING
(AUTONOMOUS)
DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTUA, Ananthapuramu Accredited by NBA, New Delhi & NAAC with 'A' grade)
Karakambadi Road, TIRUPATI – 517507
2023 – 2026

Dr. V. Lakshmi Devi Ph.D.
Head of the Department

- **Name of the Trainee** :KALLE NAGA VARDHAN
- **Name of the Company** :ByteXL TechED Pvt. Ltd
- **Name of the Supervisor/Guide** :THARUNI S
- **Title of Report** :FLIGHT BOOKING SYSTEM USING DSA IN C
- **Field of Training** : Programming Technologies
- **Area of the project** : C Programming using DSA

Abstract

The Flight Booking System deployed in C is an easy and efficient program with the purpose of handling flight timetables and passenger reservations via linked lists. The system is equipped with the features of inserting, modifying, deleting, and searching for flights, and reserving seats dynamically for passengers.

The program performs data storage and retrieval effectively with two linked lists:

Flight List – Stores flight information such as flight number, destination, departure time, and seats available.

Booking List – Keeps passenger information, such as booked flight number, passenger name, and seated seat number.

The system provides real-time seat availability updates, supporting users to book seats, change flight information, and display passenger bookings dynamically. Dynamic memory allocation increases flexibility, and the system is well-fit for real-life airline reservation applications. This project illustrates fundamental data structures like linked lists, memory management, and efficient search and update mechanisms in C programming. It may be extended further with file handling, a graphical interface, or database integration to have more advanced features.

Introduction :

FLIGHT BOOKING SYSTEM

This is a Flight Booking System in C that is intended to operate efficiently in managing flights and passenger bookings through the use of linked lists. It enables users to add, update, delete, and search for flights, as well as book seats for passengers dynamically. The system updates seat availability in real-time and keeps a record of all booked passengers.

Purpose of the Program

The primary goal of this program is to replicate a basic airline reservation system with dynamic memory allocation and linked lists. It presents an organized method of handling flight schedules and bookings of passengers without having to utilize static arrays, thus being adaptable and scalable.

Key Features

1. Flight Management

Insert new flights with information like flight number, destination, departure time, and seats available.

Search for a flight based on their flight number.

Update flight information if schedules or seat availability are modified.

Remove flights that are no longer available.

2. Booking System

Passengers can reserve seats on a chosen flight.

The system automatically adjusts the number of available seats.

Bookings are kept in a linked list with information such as passenger name, seat number, and flight number.

Shows a list of reserved passengers for each flight.

3. Data Management Using Linked Lists

The system has two linked lists:

Flight List: Stores details of all flights.

Booking List: Stores details of all passenger bookings.

This ensures efficient memory usage and easy modification of records.

How the Program Works

1. Adding Flights:
The user inputs the number of flights and their details (flight number, destination, departure time, and available seats).
These flights are stored in a linked list.

2. Displaying Flights:
The system displays the list of available flights with flight numbers, destinations, departure times, and seats available.

3. Booking a Seat:
The user inputs a flight number and passenger name.
If there are available seats, a seat number is allocated, and the booking is stored.
The count of available seats is decreased accordingly.

4. Updating Flight Details:
The user can update flight details like destination, departure time, and seat availability.

5. Deleting a Flight:
A flight can be deleted from the system, and all related bookings are erased.

6. Displaying Bookings:
The system displays all passengers who have booked seats for a given flight, along with seat numbers.

SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS:

Hardware - intel i3
Speed - 2.1 GHz
RAM - 4GB
Hard Disk - 100 GB SSD
Floppy Drive - 2.88 MB
Key Board - Standard Windows Keyboard
Mouse - Two or Three-Button Mouse

SOFTWARE REQUIREMENTS:

Operating System: Windows 10, 11
Technology: C
Compiler: byteXL TechEd editor(https://bytexl.app/editor).

Code

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// Structure for a Flight node
typedef struct Flight {
    int flightNumber;
    char destination[50];
    char departure[50];
    int seatsAvailable;
    struct Flight* next;
} Flight;

// Structure for a Booking node
typedef struct Booking {
    int flightNumber;
    char passengerName[50];
    int seatNumber;
    struct Booking* next;
} Booking;

// Head pointers for the linked lists
Flight* flightHead = NULL;
Booking* bookingHead = NULL;

// Function to create a new flight node
Flight* createFlight(int flightNumber, char* destination, char* departure, int
seatsAvailable) {
    Flight* newFlight = (Flight*)malloc(sizeof(Flight));
    newFlight->flightNumber = flightNumber;
    strcpy(newFlight->destination, destination);
    strcpy(newFlight->departure, departure);
    newFlight->seatsAvailable = seatsAvailable;
    newFlight->next = NULL;
    return newFlight;
}

// Function to add a flight to the linked list
void addFlight(int flightNumber, char* destination, char* departure, int seatsAvailable) {
    Flight* newFlight = createFlight(flightNumber, destination, departure, seatsAvailable);
    newFlight->next = flightHead;
    flightHead = newFlight;
}
```

```
// Function to search for a flight by flight number
Flight* searchFlight(int flightNumber) {
    Flight* current = flightHead;
    while (current != NULL) {
        if (current->flightNumber == flightNumber) {
            return current;
        }
        current = current->next;
    }
    return NULL;
}

// Function to create a new booking node
Booking* createBooking(int flightNumber, char* passengerName, int seatNumber) {
    Booking* newBooking = (Booking*)malloc(sizeof(Booking));
    newBooking->flightNumber = flightNumber;
    strcpy(newBooking->passengerName, passengerName);
    newBooking->seatNumber = seatNumber;
    newBooking->next = NULL;
    return newBooking;
}

// Function to add a booking to the linked list
void addBooking(int flightNumber, char* passengerName, int seatNumber) {
    Booking* newBooking = createBooking(flightNumber, passengerName, seatNumber);
    newBooking->next = bookingHead;
    bookingHead = newBooking;
}

// Function to book a seat
void bookSeat(int flightNumber, char* passengerName) {
    Flight* flight = searchFlight(flightNumber);
    if (flight == NULL) {
        printf("Flight not found.\n");
        return;
    }
    if (flight->seatsAvailable == 0) {
        printf("No seats available on this flight.\n");
        return;
    }

    flight->seatsAvailable--;
    int seatNumber = 100 - flight->seatsAvailable; // Assume a fixed max seats of 50
    addBooking(flightNumber, passengerName, seatNumber);

    printf("Booking successful! Seat number: %d\n", seatNumber);
```

```
}

// Function to display all flights
void displayFlights() {
    Flight* current = flightHead;
    printf("Flight Number\tDestination\tDeparture\tSeats Available\n");
    while (current != NULL) {
        printf("%d\t\t %s\t\t%s \t\t%d\n", current->flightNumber, current->destination,
current->departure, current->seatsAvailable);
        current = current->next;
    }
}

// Function to display bookings for a specific flight
void displayBookings(int flightNumber) {
    Booking* current = bookingHead;
    printf("Passenger Name\tSeat Number\n");
    while (current != NULL) {
        if (current->flightNumber == flightNumber) {
            printf("%s\t\t%d\n", current->passengerName, current->seatNumber);
        }
        current = current->next;
    }
}
// Function to update flight details
void updateFlight(int flightNumber, char* destination, char* departure, int
seatsAvailable) {
    Flight* flight = searchFlight(flightNumber);
    if (flight == NULL) {
        printf("Flight not found.\n");
        return;
    }
    strcpy(flight->destination, destination);
    strcpy(flight->departure, departure);
    flight->seatsAvailable = seatsAvailable;
    printf("Flight details updated successfully.\n");
}

// Function to delete a flight
void deleteFlight(int flightNumber) {
    Flight* current = flightHead;
    Flight* previous = NULL;
    while (current != NULL) {
        if (current->flightNumber == flightNumber) {
            if (previous == NULL) {
                flightHead = current->next;
```

```
            } else {
                previous->next = current->next;
            }
            free(current);
            printf("Flight deleted successfully.\n");
            return;
        }
        previous = current;
        current = current->next;
    }
    printf("Flight not found.\n");
}

int main() {
    int n;
    scanf("%d\n", &n);
    int flno, flseat;
    char fldest[50];
    char fltime[30];
    for (int i = 1; i <= n; i++) {
        scanf("%d\n", &flno);
        scanf("%s\n", fldest);
        scanf("%s\n", fltime);
        scanf("%d\n", &flseat);
        addFlight(flno, fldest, fltime, flseat);
    }

    displayFlights();
    bookSeat(101, "Raghava sai");
    bookSeat(103, "Naga vardhan");
    bookSeat(103, "Arun kumar");
    bookSeat(103, "Baba fakruddin");

    displayFlights();
    // Example of update and delete
    updateFlight(101, "HYDERABAD", "3:30", 40);
    displayFlights();

    deleteFlight(104); // Delete a non-existent flight
    deleteFlight(101); // Delete an existing flight
    displayFlights();

    // Example of displaying bookings
    displayBookings(103);
     return 0;
}
```

Code Explanation

- Data Structures:
- Flight: Represents a flight with its number, destination, departure time, available seats, and a pointer to the next flight.
- Booking: Represents a booking with the flight number, passenger name, seat number, and a pointer to the next booking.
- flightHead and bookingHead: Pointers to the first flight and booking nodes, respectively.
- Flight Management Functions:
- createFlight(): A function that makes a new Flight node and sets its members.
- addFlight(): Inserts a new flight at the front of the flightHead linked list.
- searchFlight(): Finds a flight using its number and returns a pointer to the Flight node or NULL if not existing.
- displayFlights(): Displays the information of all flights in the linked list.
- updateFlight(): Changes the destination, departure, and seats available for a given flight.
- deleteFlight(): Deletes a flight from the linked list.
- Booking Management Functions:
- createBooking(): Inserts a new Booking node.
- addBooking(): Adds a new booking at the head of the bookingHead linked list.
- bookSeat(): Books a seat on a flight, updates the available seats, and adds a booking to the bookingHead list. It also allocates a seat number.
- displayBookings(): Prints the information of all bookings for a given flight.
- main() Function:
- Reads the number of flights from the user.
- Reads flight details (number, destination, departure, seats) from the user and adds them to the flightHead linked list.
- Displays all flights.
- Books seats for several passengers on flights 101 and 103.
- Displays the updated flight list.
- Updates flight 101.
- Displays the flight list again.
- Deletes flight 104, and then 101.
- Displays the flight list after deletions.
- Displays the bookings for flight 103.

Output

Flight Number	Destination	Departure	Seats Available
103	BENGALURU	21:00	56
104	DELHI	2:00	75
101	MUMBAI	15:00	65

Booking successful! Seat number: 36

Booking successful! Seat number: 45

Booking successful! Seat number: 46

Booking successful! Seat number: 47

Flight Number	Destination	Departure	Seats Available
103	BENGALURU	21:00	53
104	DELHI	2:00	75
101	MUMBAI	15:00	64

Flight details updated successfully.

Flight Number	Destination	Departure	Seats Available
103	BENGALURU	21:00	53
104	DELHI	2:00	75
101	HYDERABAD	3:30	40

Flight deleted successfully.

Flight deleted successfully.

Flight Number	Destination	Departure	Seats Available
103	BENGALURU	21:00	53

Passenger Name	Seat Number
Baba fakruddin	47
Arun kumar	46
Naga vardhan	45


Thank
you!