/**********ASSIGNMENT_9**********
****

There are flight paths between cities. If there is a flight between city A and city B then there is an edge between the cities. The cost of the edge can be the time that flight takes to reach city B from A or the amount of fuel used for the journey. Represent this as a graph. The node can be represented by the airport name or name of the city. Use adjacency list representation of the graph or use adjacency matrix representation of the graph. Check whether the graph is connected or not.

Name: Om Sarangdhar Lahamge
Roll No.: 31 Batch: S2

****
***/

```
#include<stdio.h>

#define MAX 100

int graph[MAX][MAX];  // Adjacency matrix
int visited[MAX];     // To track visited cities
int n;                // Number of cities

// DFS function to check connectivity
void dfs(int city) {
    visited[city] = 1;
    for (int i = 0; i<n; i++) {
        if (graph[city][i] != 0&&!visited[i]) {
            dfs(i);
        }
    }
}

int main() {
    int i, j, edges, city1, city2, cost;

    printf("Enter number of cities:");
    scanf("%d",&n);

    // Initialize graph
    for (i = 0; i<n; i++) {
        for (j = 0; j<n; j++) {
```

```
            graph[i][j] = 0;
        }
    }

    printf("Enter number of flight paths:");
    scanf("%d",&edges);

    printf("Enter flight paths (city1 city2 cost):\n");
    for (i = 0; i<edges; i++) {
        scanf("%d %d %d",&city1,&city2,&cost);
        // assuming 0-based indexing for cities
        graph[city1][city2] = cost;
        graph[city2][city1] = cost;  // for undirected graph
    }

    // Start DFS from city 0
    dfs(0);

    // Check if all cities are visited
    int connected = 1;
    for (i = 0; i<n; i++) {
        if (!visited[i]) {
            connected = 0;
            break;
        }
    }

    if (connected)
        printf("The graph is connected. All cities are reachable.\n");
    else
        printf("The graph is NOT connected. Some cities are not reachable.\n");

    return 0;
}
```

OUTPUT:

```
student@student-OptiPlex-3000:~$ gedit ass9.c
student@student-OptiPlex-3000:~$ gcc ass9.c -o ass9
student@student-OptiPlex-3000:~$ ./ass9
```

Enter number of cities: 4
Enter number of flight paths: 3
Enter flight paths (city1 city2 cost):
0 1 10
1 2 20
2 3 30
The graph is connected. All cities are reachable.