Experiment No. 7

Name : Aliasghar Masood SY IT
Roll No : 33

PROGRAM:

DSF

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

#define MAX_VERTICES 20

int source, V, E, visited[MAX_VERTICES], G[MAX_VERTICES]
[MAX_VERTICES];

void DFS(int vertex) {
   printf("%d -> ", vertex + 1);
   visited[vertex] = 1;

   for (int j = 0; j < V; j++) {
      if (G[vertex][j] == 1 && !visited[j]) {
         DFS(j);
      }
   }
}

int main() {
   int v1, v2;

   printf("\t\t\tGraphs\n");
   printf("Enter the number of edges: ");
   scanf("%d", &E);

   printf("Enter the number of vertices: ");
   scanf("%d", &V);
```

```
    for (int i = 0; i < V; i++) {
        for (int j = 0; j < V; j++) {
            G[i][j] = 0;
        }
    }

    for (int i = 0; i < E; i++) {
        printf("Enter edge (format: V1 V2): ");
        scanf("%d %d", &v1, &v2);

        if (v1 >= 1 && v1 <= V && v2 >= 1 && v2 <= V) {
            G[v1 - 1][v2 - 1] = 1;
            G[v2 - 1][v1 - 1] = 1; // If your graph is undirected
        } else {
            printf("Invalid input. Please enter valid vertices.\n");
            i--; // Decrement 'i' to re-enter the edge
        }
    }

    printf("Adjacency Matrix:\n");
    for (int i = 0; i < V; i++) {
        for (int j = 0; j < V; j++) {
            printf("%d ", G[i][j]);
        }
        printf("\n");
    }

    printf("Enter the source vertex: ");
    scanf("%d", &source);

    if (source >= 1 && source <= V) {
        printf("DFS Traversal starting from vertex %d:\n", source);
        DFS(source - 1);
    } else {
        printf("Invalid source vertex.\n");
    }

    return 0;
}
```

OUTPUT:

```
itl4@itadmin:~$ gcc qasim7.c
itl4@itadmin:~$ ./a.out
                        Graphs
Enter the number of edges: 9
Enter the number of vertices: 7
Enter edge (format: V1 V2): 4 5
Enter edge (format: V1 V2): 7 3
Enter edge (format: V1 V2): 8 6
Invalid input. Please enter valid vertices.
Enter edge (format: V1 V2): 9 2
Invalid input. Please enter valid vertices.
Enter edge (format: V1 V2): 1 4
Enter edge (format: V1 V2): 3 3
Enter edge (format: V1 V2): 2 2
Enter edge (format: V1 V2): 4 4
Enter edge (format: V1 V2): 5 5
Enter edge (format: V1 V2): 6 6
Enter edge (format: V1 V2): 7 7
Adjacency Matrix:
0 0 0 1 0 0 0
0 1 0 0 0 0 0
0 0 1 0 0 0 1
1 0 0 1 1 0 0
0 0 0 1 1 0 0
0 0 0 0 0 1 0
0 0 1 0 0 0 1
Enter the source vertex: 6
DFS Traversal starting from vertex 6:
6 -> itl4@itadmin:~$
```

BSF

PROGRAM:

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

#define MAX_VERTICES 20

int a[MAX_VERTICES][MAX_VERTICES], q[MAX_VERTICES],
visited[MAX_VERTICES], n, f = -1, r = -1;

void bfs(int v) {
    int i;
    for (i = 0; i < n; i++) {
        if (a[v][i] != 0 && visited[i] == 0) {
```

```
            r = r + 1;
            q[r] = i;
            visited[i] = 1;
            printf("%d ", i);
        }
    }
    f = f + 1;
    if (f <= r) {
        bfs(q[f]);
    }
}

int main() {
    int v, i, j;
    printf("\n\t\t\tBreadth-First Search (BFS)\n");
    printf("\nEnter the number of vertices: ");
    scanf("%d", &n);

    for (i = 0; i < n; i++) {
        visited[i] = 0;
        printf("\nEnter graph data in matrix form for vertex %d:\n", i);
        for (j = 0; j < n; j++) {
            scanf("%d", &a[i][j]);
        }
    }

    printf("\nEnter the starting vertex: ");
    scanf("%d", &v);

    if (v >= 0 && v < n) {
        f = r = 0;
        q[r] = v;
        visited[v] = 1;
        printf("BFS Traversal starting from vertex %d:\n", v);
        printf("%d ", v);
        bfs(v);

        if (r != n - 1) {
            printf("\nBFS not possible\n");
        }
    } else {
```

```
        printf("Invalid starting vertex.\n");
    }

    return 0;
}
```

OUTPUT: