```
#include <iostream>
#include <stdlib.h>
using namespace std;

int cost[10][10], i, j, k, n, m, qu[10], front = 0, rear = 0, v, visited[10], visit[10];
int stk[10], top = 0, visit1[10], visited1[10];

int main()
{
   cout << "Enter number of vertices: ";
   cin >> n;
   cout << "Enter number of edges: ";
   cin >> m;

   // Initialize adjacency matrix
   for (i = 0; i < n; i++) {
      for (j = 0; j < n; j++) {
         cost[i][j] = 0;
      }
   }

   // Read edges
   cout << "\nEDGES:\n";
   for (k = 0; k <= m - 1; k++) {
      cout << "Enter edge " << k << " (in the format: vertex1 vertex2): ";
      cin >> i >> j;
      cost[i][j] = 1;
      cost[j][i] = 1;  // Because the graph is undirected
   }

   // Display adjacency matrix
   cout << "The adjacency matrix of the graph is:\n";
   for (i = 0; i < n; i++) {
      for (j = 0; j < n; j++) {
         cout << " " << cost[i][j];
      }
      cout << endl;
   }

   // BFS
   cout << "Enter initial vertex for BFS: ";
   cin >> v;
   cout << "The BFS of the Graph is: \n";
   cout << v << " ";
```

```
    visited[v] = 1;
    k = 0;
    while (k < n) {
        for (j = 0; j < n; j++) {
            if (cost[v][j] != 0 && visited[j] != 1 && visit[j] != 1) {
                visit[j] = 1;
                qu[rear++] = j;
            }
        }
        v = qu[front++];
        cout << v << " ";
        k++;
        visit[v] = 0;
        visited[v] = 1;
    }
    cout << endl;

    // DFS
    cout << "Enter initial vertex for DFS: ";
    cin >> v;
    cout << "The DFS of the Graph is: \n";
    cout << v << " ";
    visited1[v] = 1;
    k = 0;
    stk[top++] = v;
    while (k < n) {
        for (j = n - 1; j >= 0; j--) {
            if (cost[v][j] != 0 && visited1[j] != 1 && visit1[j] != 1) {
                visit1[j] = 1;
                stk[top++] = j;
            }
        }
        v = stk[--top];
        cout << v << " ";
        k++;
        visit1[v] = 0;
        visited1[v] = 1;
    }
    cout << endl;
    return 0;
}
```