

## **PROBLEM STATEMENT 2**

/\* Depth First Search (DFS):

Application: Web crawlers use DFS to explore web pages systematically, following links and indexing content for search engines. Write a simple program to index web pages using Depth First Search (DFS). The program should simulate a web graph where pages are represented as nodes and hyperlinks as edges.\*/

```
#include <iostream>
using namespace std;
class Graph {
    int vertices;           //stores number of vertices
    int adjMatrix[10][10]; //adjacency matrix for graph
    int visited[10];        //mark visited nodes

public:
    //Function to read graph details
    void readGraph() {
        int edges;
        cout<<"\nEnter number of vertices: ";
        cin>>vertices;

        //initialize matrix and visited array to 0
        for (int i = 1; i <= vertices; i++) {
            for (int j = 1; j <= vertices; j++) {
                adjMatrix[i][j] = 0;
            }
            visited[i] = 0;
        }

        cout << "Enter number of edges: ";
        cin >> edges;

        cout << "Enter edges (u v):\n";
        for (int k = 0; k < edges; k++) {
            int u, v;
            cin >> u >> v;
            // put 1 in matrix to show that there is an edge
            adjMatrix[u][v] = 1;
            adjMatrix[v][u] = 1;    //because the graph is undirected
        }
    }

    //Function for Depth First Search
    void dfs(int node) {
```

```

    cout << node << " ";    //print the current node
    visited[node] = 1;      //mark it as visited

    //visit all connected vertices of this node
    for (int i = 1; i <= vertices; i++) {
        //if there is a connection and that node is not yet visited
        if (adjMatrix[node][i] == 1 && visited[i] == 0) {
            dfs(i);          //call dfs again for that node
        }
    }
}

//Function to start DFS traversal
void performDFS() {
    int start;
    cout << "Enter starting vertex: ";
    cin >> start;
    cout << "DFS traversal: ";
    dfs(start);
    cout << endl;
}

};

int main() {
    Graph g;
    cout<<"Name:Manasvi Lunawat  PRN:B24CE1136"<<endl;
    g.readGraph();
    g.performDFS();    //perform DFS traversal
    return 0;
}

```

## **OUTPUT:**

```

Name:Manasvi Lunawat  PRN:B24CE1136

Enter number of vertices: 5
Enter number of edges: 5
Enter edges (u v):
1 2
1 3
2 4
3 5
4 5
Enter starting vertex: 1
DFS traversal: 1 2 4 5 3

```