**GROUP C**

**Represent a given graph using adjacency matrix/list to perform DFS and using adjacency**

**list to perform BFS. Use the map of the area around the college as the graph. Identify the**

**prominent land marks as nodes and perform DFS and BFS on that.**

#include <iostream>

#include <stdlib.h>

using namespace std;

int cost[10][10], i, j, k, n, qu[10], front, rear, v, visit[10], visited[10];

int stk[10], top, visit1[10], visited1[10];

int main()

{

int m;

cout << "Enter number of vertices : ";

cin >> n;

cout << "Enter number of edges : ";

cin >> m;

cout << "\nEDGES :\n";

for (k = 1; k <= m; k++)

{

cin >> i >> j;

cost[i][j] = 1;

cost[j][i] = 1;

}

cout << "The adjacency matrix of the graph is : " << endl;

for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

{

cout << " " << cost[i][j];

}

cout << endl;

}

cout << "Enter initial vertex : ";

cin >> v;

cout << "The BFS of the Graph is\n";

cout << v<<endl;

visited[v] = 1;

k = 1;

while (k < n)

{

for (j = 1; 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<<"Enter initial vertex : ";

cin >> v;

cout << "The DFS of the Graph is\n";

cout << v<<endl;

visited[v] = 1;

k = 1;

while (k < n)

{

for (j = n; j >= 1; j--)

if (cost[v][j] != 0 && visited1[j] != 1 && visit1[j] != 1)

{

visit1[j] = 1;

stk[top] = j;

top++;

}

v = stk[--top];

cout << v << " ";

k++;

visit1[v] = 0;

visited1[v] = 1;

}

return 0;

}

**OUTPUT:**

**Enter number of vertices : 4**

**Enter number of edges : 4**

**EDGES :**

**ab**

**The adjacency matrix of the graph is :**

**1 0 0 0**

**0 0 0 0**

**0 0 0 0**

**0 0 0 0**

**Enter initial vertex : The BFS of the Graph is**

**0**

**0 0 0**

**Enter initial vertex : The DFS of the Graph is**

**0**

**0 0**