Name: Surwade Trisharan Rajesh

Roll no.: 48

//Write a program to implement breadth first traversal.

```
#include<stdio.h>
#include<conio.h>
#include<iostream>
using namespace std;
class BFT
{
private:
    int matrix[50][50], n;
    int visited[50];
    int q[50], front, rear;
public:
    void getdata();
    void bft(int v);

};
void BFT::getdata()
{
    int i, j;
    front = rear = 1;
    cout << "\n Enter the number of the nodes";
    cin >> n;
    cout << "\n Enter the matrix";
    for (i = 1; i <= n; i++)
    {
        for (j = 1; j <= n; j++)
        {
            cin >> matrix[i][j];
        }
    }
    for (i = 1; i <= n; i++)
    {
        visited[i] = 0;

    }
}
void BFT::bft(int v)
{
```

```
    int i, t;
    for (i = 1; i <= n; i++)
    {
        visited[i] = 0;
    }
    int u = v;
    t = '\0';
    visited[v] = 1;
    cout << v << "\t";
    do
    {
        for (int w = 1; w <= n; w++)
        {
            if (matrix[u][w] == 1)
            {
                if (visited[w] == 0)
                {
                    q[rear++] = w;
                    visited[w] = 1;
                    t = t + (u, w);
                    cout << w << "\t";


                }
            }
        }
        if (front == rear)
        {
            break;
        }
        u = q[front++];
    } while (1);
}
int main()
{
    int v;
    BFT b;
    b.getdata();
    cout << "\n Enter the starting node:";
    cin >> v;
    cout << "\n Visited node using BFT:";
```

```
    b.bft(v);
    return 0;
}
```

Output:

Enter the number of the nodes4

Enter the matrix1 0 1 0

0 0 1 1

0 1 0 1

1 1 0 0

//Write a program to implement depth first traversal.

```
#include<iostream>
#include<conio.h>
#include<process.h>

using namespace std;
int Visited[20], v, a[20][20], n, i;
class DFS
{
public:
    void getdata();
    void dfs(int);
    void dft();
}d;
void DFS::getdata()
{
    int i, j;
    cout << "\n Enter the vertices : ";
    cin >> n;
    cout << "\n Enter the Adjacency matrix";
    for (i = 1; i <= n; i++)
    {
        for (j = 1; j <= n; j++)
        {
            cin >> a[i][j];
        }
    }
```

```
}
void DFS::dfs(int v)
{
    Visited[v] = 1;
    cout << v << "\t";
    for (int w = 1; w <= n; w++)
    {
        if (a[v][w] == 1)
            if (Visited[w] == 0)
                dfs(w);
    }
}
void DFS::dft()
{
    for (int i = 1; i <= n; i++)
        Visited[i] = 0;
    for (int i = 1; i < n; i++)
        if(Visited[i]==0)
            dfs(i);
}
int main()
{
    d.getdata();
    cout << "\n DFS order of nodes is  :  ";
    d.dft();
    return 0;
}
```

Output:

```
 Enter the vertices: 8

 Enter the Adjacency matrix0 1 1 0 0 0 0 0
1 0 0 1 1 0 0 0
0 0 0 0 0 1 1 0
0 1 0 0 0 0 0 1
0 1 0 0 0 0 0 1
0 0 1 0 0 0 0 1
0 0 1 0 0 0 0 1
0 0 0 1 1 1 1 0

 DFS order of nodes is:  1    2       4       8       5       6       3
7
```