Practical Graph

Name : Abhishek Gupta   UID No : 2019450017

Program to display Adjacency Matrix using Graph

Code :

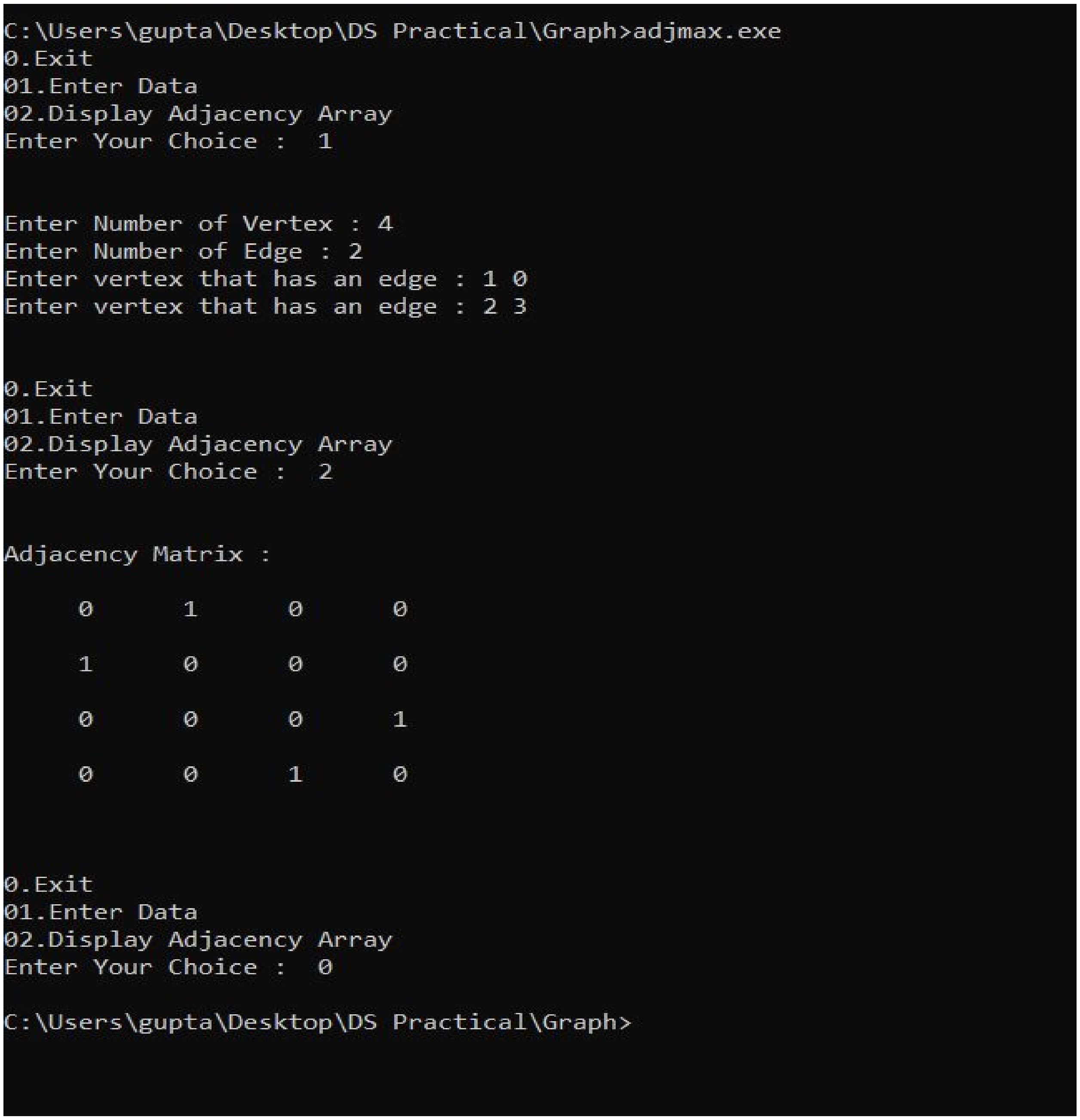
```
#include <iostream>
#include <stdlib.h>

using namespace std;

class AdjacencyMatrix
{
    private :
        int adjarr[10][10];
        int vertex,edge,choice,edge1,edge2;
    public :
    AdjacencyMatrix()
    {
        for(int j=0;j<vertex;j++)
        {
            for(int z=0;z<vertex;z++)
            {
                adjarr[j][z]=0;
            }
        }
    }

    void get()
    {
            do
            {
```

Practical Graph

Name : Abhishek Gupta   UID No : 2019450017

```
                cout<<"0.Exit\n01.Enter Data \n02.Display 
Adjacency Array \n";
                cout<<"Enter Your Choice : "<<" ";
                cin>>choice;
                switch(choice)
                {
                    case 0:
                    break;

                    case 1:
                        EnterData();
                        break;

                    case 2:
                        display();
                        break;

                    default:
                        cout<<"invalid input"<<endl<<endl;
                }
            }while(choice!=0);
    }

    void EnterData()
    {
        cout<<endl<<endl;
        cout<<"Enter Number of Vertex : ";
        cin>>vertex;
        cout<<"Enter Number of Edge : ";
        cin>>edge;
        for(int i=0;i<edge;i++)
        {
            cout<<"Enter vertex that has an edge : ";
            cin>>edge1;
            cin>>edge2;
            adjarr[edge1][edge2]=1;
            adjarr[edge2][edge1]=1;
        }
        cout<<endl<<endl;
    }

    void display()
    {
        cout<<endl<<endl;
        cout<<"Adjacency Matrix : ";
        cout<<endl<<endl;
        for(int j=0;j<vertex;j++)
        {
            for(int z=0;z<vertex;z++)
            {
                cout<<"     "<<adjarr[j][z]<<" ";
            }
            cout<<endl<<endl;
        }
        cout<<endl<<endl;
    }

};
int main()
{
    AdjacencyMatrix d;
    d.get();
    return 0;
}
```

Output :



```
C:\Users\gupta\Desktop\DS Practical\Graph>adjmax.exe
0.Exit
01.Enter Data
02.Display Adjacency Array
Enter Your Choice :  1


Enter Number of Vertex : 4
Enter Number of Edge : 2
Enter vertex that has an edge : 1 0
Enter vertex that has an edge : 2 3


0.Exit
01.Enter Data
02.Display Adjacency Array
Enter Your Choice :  2


Adjacency Matrix :

     0      1      0      0

     1      0      0      0

     0      0      0      1

     0      0      1      0



0.Exit
01.Enter Data
02.Display Adjacency Array
Enter Your Choice :  0

C:\Users\gupta\Desktop\DS Practical\Graph>
```

Name : Abhishek Gupta UID No : 2019450017

Program to display DFS using Graph

Code :

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

using namespace std;

class DFS
{
    public :
    int cost[10][10],stk[10],visit[10],visited[10];
    int i,j,k,x,m,n,top,v,choice;

    void get()
    {
            do
            {
                cout<<endl;
                cout<<"0.Exit\n01.Enter Data \n02.Display DFS
\n03.Display Adjacency Array \n";
                cout<<"Enter Your Choice : "<<" ";
                cin>>choice;
                switch(choice)
                {
                    case 0:
                    break;

                    case 1:
```

```
                    EnterData();
                    break;

                case 2:
                    ShowDfs();
                    break;

                case 3:
                    display();
                    break;

                default:
                    cout<<"invalid input"<<endl<<endl;
            }
        }while(choice!=0);
}

void EnterData()
{
cout<<endl<<endl;
cout <<"Enter no of vertices:";
cin >> n;
cout <<"Enter no of edges:";
cin >> m;
cout <<"\nEDGES \n";
for(k=1; k<=m; k++)
{
    cin >>i>>j;
    cost[i][j]=1;
    cost[j][i]=1;
}
```

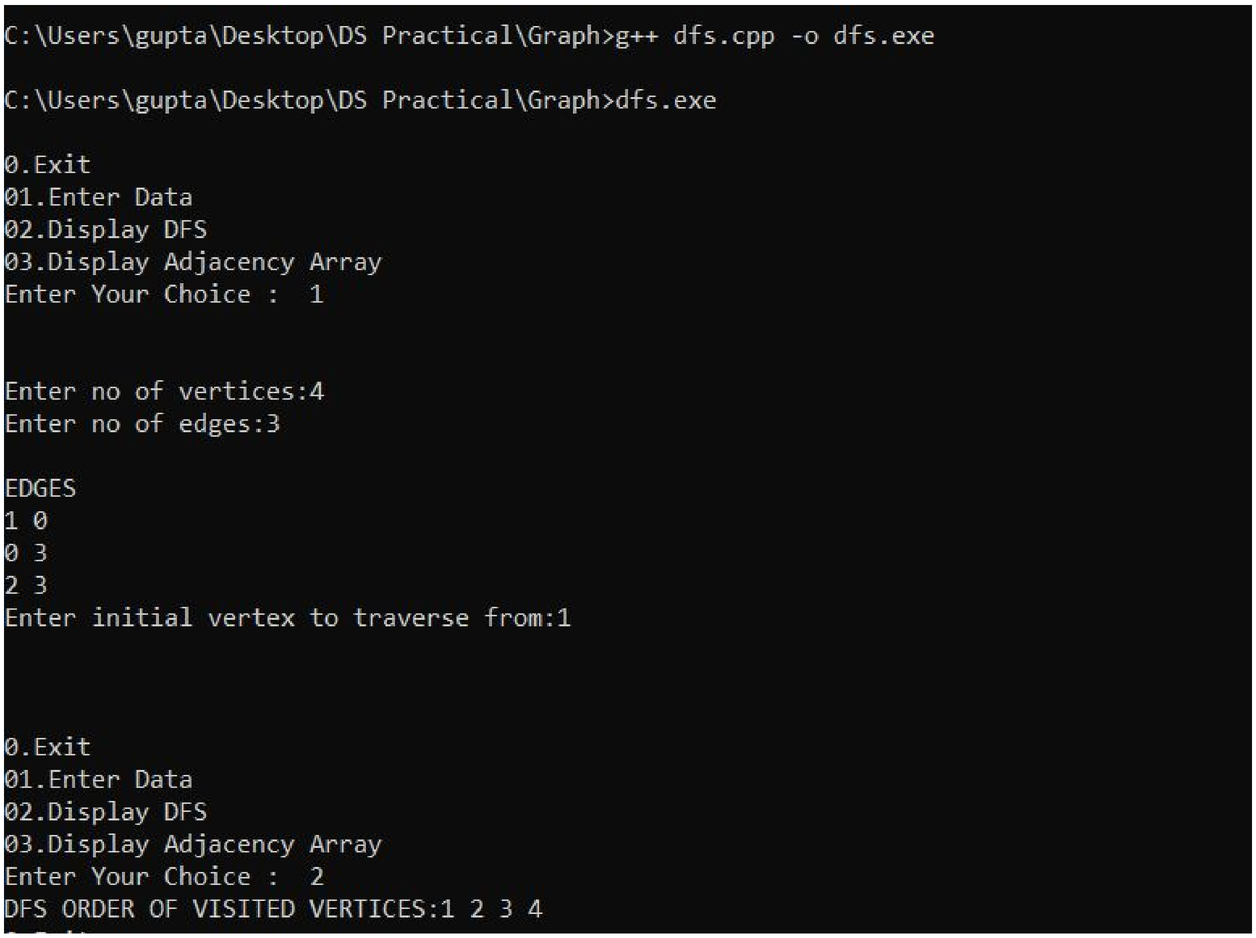
```
cout <<"Enter initial vertex to traverse from:";
cin >>v;
cout<<endl<<endl;
}

void ShowDfs()
{
cout <<"DFS ORDER OF VISITED VERTICES:";
cout << v <<" ";
visited[v]=1;
k=1;
while(k<n)
{
    for(j=n; j>=1; j--)
        if(cost[v][j]!=0 && visited[j]!=1 && visit[j]!=1)
        {
            visit[j]=1;
            stk[top]=j;
            top++;
        }
    v=stk[--top];
    cout<<v << " ";
    k++;
    visit[v]=0;
    visited[v]=1;
}
}

void display()
{
    cout<<endl<<endl;
```

```
        cout<<"Adjacency Matrix : ";
        cout<<endl<<endl;
        for(int j=0;j<n;j++)
        {
            for(int z=0;z<n;z++)
            {
                cout<<"     "<<cost[j][z]<<" ";
            }
            cout<<endl<<endl;
        }
        cout<<endl<<endl;
    }

};

int main()
{
    DFS d;
    d.get();
    return 0;
}
```

Practical Graph

Name : Abhishek Gupta **UID No : 2019450017**

Output :



```
C:\Users\gupta\Desktop\DS Practical\Graph>g++ dfs.cpp -o dfs.exe

C:\Users\gupta\Desktop\DS Practical\Graph>dfs.exe

0.Exit
01.Enter Data
02.Display DFS
03.Display Adjacency Array
Enter Your Choice :  1


Enter no of vertices:4
Enter no of edges:3

EDGES
1 0
0 3
2 3
Enter initial vertex to traverse from:1



0.Exit
01.Enter Data
02.Display DFS
03.Display Adjacency Array
Enter Your Choice :  2
DFS ORDER OF VISITED VERTICES:1 2 3 4
```

Practical Graph

Name : Abhishek Gupta **UID No : 2019450017**

Program to display BFS using Graph
Code :

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

using namespace std;

class BFS
{
    public :
    int cost[10][10],qu[10],visit[10],visited[10];
    int i,j,k,n,front,rare,v,choice,m;

    void get()
    {
            do
            {
                cout<<endl;
                cout<<"0.Exit\n01.Enter Data \n02.Display BFS
\n03.Display Adjacency Array \n";
                cout<<"Enter Your Choice : "<<" ";
                cin>>choice;
                switch(choice)
                {
                    case 0:
                    break;

                    case 1:
                        EnterData();
                        break;

                    case 2:
                        ShowDfs();
                        break;

                    case 3:
                        display();
                        break;

                    default:
                        cout<<"invalid input"<<endl<<endl;
                }
            }while(choice!=0);
    }

    void EnterData()
    {
    cout<<endl<<endl;
    cout <<"Enter no of vertices:";
    cin >> n;
    cout <<"Enter no 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 <<"Enter initial vertex to traverse from:";
```

Practical Graph

Name : Abhishek Gupta UID No : 2019450017

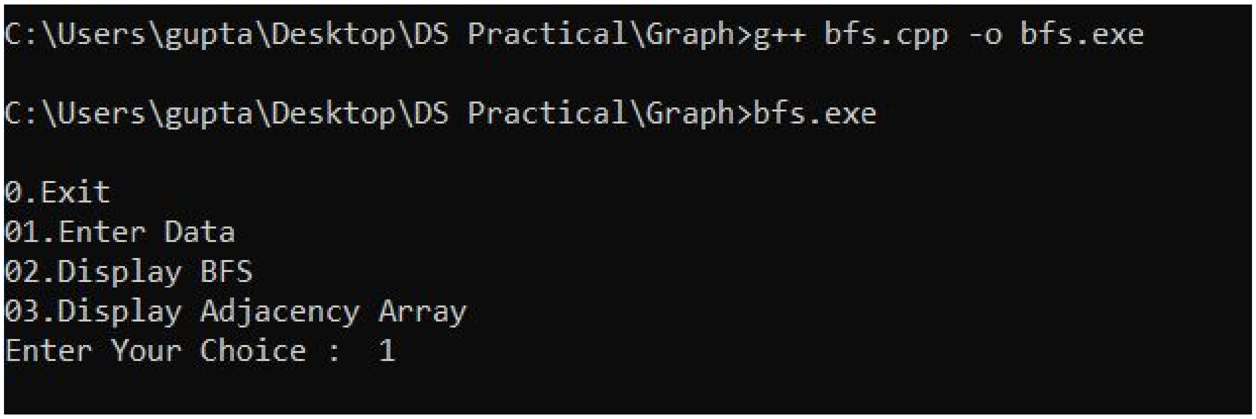
```
cin >>v;
cout<<endl<<endl;
}

void ShowDfs()
{
cout <<"Visitied vertices:";
cout <<v<<" ";
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[rare++]=j;
        }
    v=qu[front++];
    cout<<v <<" ";
    k++;
    visit[v]=0;
    visited[v]=1;
}
}

void display()
{
    cout<<endl<<endl;
    cout<<"Adjacency Matrix : ";
    cout<<endl<<endl;
```

```
        for(int j=0;j<n;j++)
        {
            for(int z=0;z<n;z++)
            {
                cout<<"     "<<cost[j][z]<<" ";
            }
            cout<<endl<<endl;
        }
        cout<<endl<<endl;
    }

};

int main()
{
    BFS b;
    b.get();
    return 0;
}
```

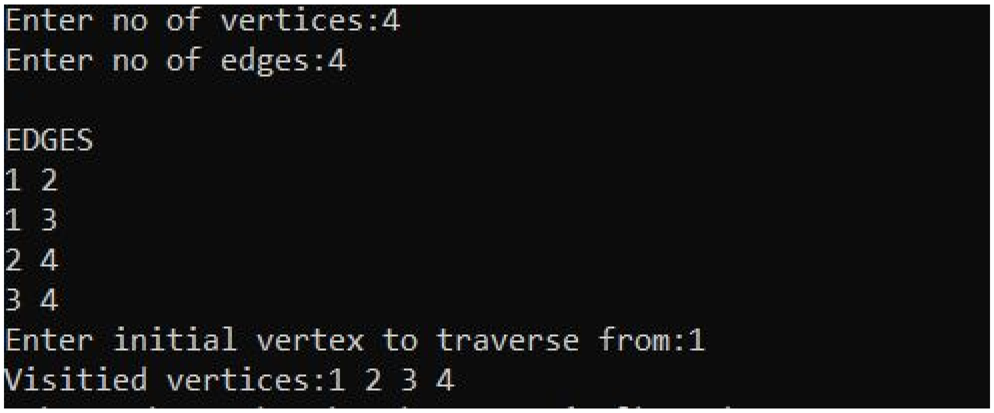
Practical Graph

Name : Abhishek Gupta **UID No : 2019450017**

Output :



```
C:\Users\gupta\Desktop\DS Practical\Graph>g++ bfs.cpp -o bfs.exe

C:\Users\gupta\Desktop\DS Practical\Graph>bfs.exe

0.Exit
01.Enter Data
02.Display BFS
03.Display Adjacency Array
Enter Your Choice :  1
```



```
Enter no of vertices:4
Enter no of edges:4

EDGES
1 2
1 3
2 4
3 4
Enter initial vertex to traverse from:1
Visitied vertices:1 2 3 4
```