American International University Bangladesh (AIUB)

Faculty of science & Technology
Department of Computer Science

LAB MANUAL DFS

CSC 2211 Algorithms

TITLE

Graph Representation Using Adjacency Matrix

```
#include <bits/stdc++.h>
using namespace std;

void countDegree(int **p, int n){

int *d = new int[n];

 for(int i=0;i<n;i++){
       d[i]=0;
    }


for(int i=0;i<n;i++){
        for(int j =0;j<n;j++){
            if(p[i][j]==1){
                d[i]++;
            }
        }
    }

    for(int i=0;i<n;i++){
        cout<<i<<" degree "<<d[i]<<endl;
    }

}


void printGraph(int **p, int n){

    for(int i=0;i<n;i++){
        for(int j =0;j<n;j++){
            cout<<p[i][j]<<" ";
        }
        cout<<endl;
    }
}


int main(){

    int node, edge;

    cin>>node>>edge;
```

```
    // Dynamic memory allocation for matrix
    int **m = new int*[node];
    for(int i=0;i<node;i++){
        m[i] = new int[node];
    }

    for(int i=0;i<node;i++){
        for(int j =0;j<node;j++){
           m[i][j]=0;
        }
    }

    int u,v;

    for(int i=0;i<edge;i++){
            cin>>u>>v;
            m[u][v]=m[v][u]=1;
    }

    printGraph(m,node);

    countDegree(m,node);



return 0;
}


/*

7 10
0 1
0 2
0 3
1 3
2 4
2 5
3 6
3 4
4 5
4 6

0 1 1 1 0 0 0
1 0 0 1 0 0 0
1 0 0 0 1 1 0
```

```
1 1 0 0 1 0 1
0 0 1 1 0 1 1
0 0 1 0 1 0 0
0 0 0 1 1 0 0

0 degree 3
1 degree 2
2 degree 3
3 degree 4
4 degree 4
5 degree 2
6 degree 2

*/
```

Graph Representation Using Adjacency List

```
#include <bits/stdc++.h>
using namespace std;

void countDegree(vector<int> p[], int n){

int *d = new int[n];

 for(int i=0;i<n;i++){
       d[i]=0;
    }


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

        d[i]=p[i].size();

    }

    for(int i=0;i<n;i++){
        cout<<i<<" degree "<<d[i]<<endl;
    }

}




void printGraph(vector<int> p[], int n){

    for(int i=0;i<n;i++){
```

```
            cout<<i<<"-->";
        for(int j =0;j<p[i].size();j++){
            cout<<p[i][j]<<" ";
        }
        cout<<endl;
    }
}


int main(){

    int node, edge;

    cin>>node>>edge;

    vector<int> m[7];


    int u,v;

    for(int i=0;i<edge;i++){
            cin>>u>>v;
            m[u].push_back(v);
            m[v].push_back(u);
    }

    printGraph(m,node);
    countDegree(m,node);


return 0;
}


/*

7 10
0 1
0 2
0 3
1 3
2 4
2 5
3 6
3 4
4 5
4 6
```

```
0-->1 2 3
1-->0 3
2-->0 4 5
3-->0 1 6 4
4-->2 3 5 6
5-->2 4
6-->3 4

0 degree 3
1 degree 2
2 degree 3
3 degree 4
4 degree 4
5 degree 2
6 degree 2

*/
```