**American International University Bangladesh (AIUB)**

****

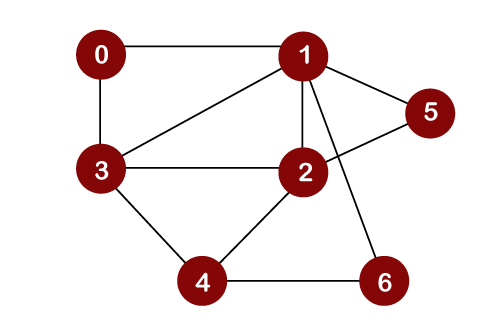
**Faculty of science & Technology**

**Department of Computer Science**

**LAB MANUAL BFS**  
CSC 2211 Algorithms

|  |
| --- |
| **TITLE** |

**BFS Algorithm**



/\*

7 11

0 1

0 3

1 3

2 3

4 3

1 2

4 2

4 6

1 6

2 5

1 5

\*/

#include<bits/stdc++.h>

using namespace std;

#define MX 100

vector<int> graph[MX];

int visited[MX];

int d[MX];

void bfs(int source){

queue<int> q;

q.push(source);

visited[source]=1;

d[source]=0;

while(!q.empty()){

int v=q.front();

q.pop();

cout<<v<<" ";

for(int i=0;i<graph[v].size();i++){

int next = graph[v][i];

if(visited[next]==0){

q.push(next);

visited[next]=1;

d[next] = d[v]+1;

}

}

}

}

int main(){

int node,edge,u,v;

cin>>node>>edge;

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

cin>>u>>v;

graph[u].push\_back(v);

graph[v].push\_back(u);

}

bfs(0);

cout<<"From 0 distance of ";

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

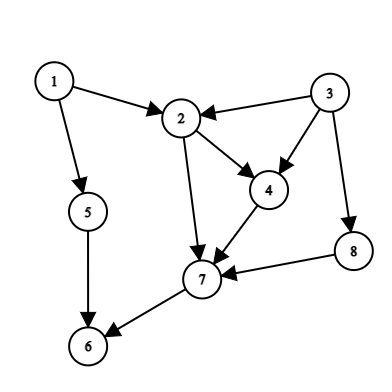
cout<<i<<" is "<<d[i]<<endl;

}

return 0;

}

**Topological Sort**

****

/\*

8 11

1 2

1 5

5 6

7 6

2 7

2 4

3 2

3 4

3 8

4 7

8 7

\*/

#include<bits/stdc++.h>

using namespace std;

#define MX 100

vector<int> graph[MX];

int visited[MX];

vector<int> result;

void dfs(int source){

visited[source]=1;

for(int i=0;i<graph[source].size();i++){

int next = graph[source][i];

if(visited[next]==0){

dfs(next);

}

}

result.push\_back(source);

}

int main(){

int node,edge,u,v;

cin>>node>>edge;

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

cin>>u>>v;

graph[u].push\_back(v);

}

for(int i=1;i<=node;i++){

if(visited[i]==0){

dfs(i);

}

}

reverse(result.begin(),result.end());

cout<<"Topological Order"<<endl;

for(int i=0;i<result.size();i++){

cout<<result[i]<<" ";

}

return 0;

}