

**19BIT0292**

**Bhaumik Tandan**

**DIGITAL ASSIGNMENT-4**

**DATA STRUCTURES**

**AND**

**ALGORITHMS**

**LABORATORY**

CSE2011

L57+L58

**Q1)** Consider the following Graph G= (V, E). Write an algorithm to implement the depth first search for the given graph and implement the same in C language. Print the results of DFS and also the adjacency matrix obtained. Calculate time complexity of algorithm developed.

**CODE**

#include<stdio.h>

#define MAX 10

int graph[MAX][MAX],v[MAX],n;

void DFS(int i)

{

printf("\n%c",i+'A');

v[i]=1;

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

if(!v[j] && graph[i][j]==1)

DFS(j);

}

void print\_adj()

{

printf("\nThe adjecency matrix of the graph is:-\n");

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

{

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

printf("%d ",graph[i][j]);

printf("\n");

}

}

void take\_input()

{

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

{

int k;

printf("Enter the number nodes connected with %c: ",i+'A');

scanf("%d",&k);

printf("Enter the nodes connected with %c: ",i+'A');

for(int j=0;j<k;j++)

{

char node;

scanf("%c",&node);

scanf("%c",&node);

graph[i][node-'A']=1;

}

}

}

main()

{

printf("Enter number of vertices: ");

scanf("%d",&n);

take\_input();

print\_adj();

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

v[i]=0;

DFS(0);

}

The time complexity of the algorithm is **O(n+e)**

where v is the number of vertices and e is the number of edges.

If we consider the printing of adjacency matrix as the part of the problem then it can be **O(n^n)**

because we are iterating to all the arrays in graph,

which has total number of element equal to the number of vertices in the graph.

**OUTPUT**

