

**DATA STRUCTURES LAB**

**WEEK-6**

**Name**  : **ABHISHEK KUMAR JHA**

**Roll no** : **19R21A05C2**

**Date**  : **6/01/2021**

**PROBLEM STATEMENT**:

Write a C program to implement the following Graph Traversal Techniques:

a)BFS b)DFS

**CODE:**

**BREADTH FIRST TREE:**

#include<stdio.h>

#define size 10

int q[size],visited[size],i,j,n,x,front=-1,rear=-1,a[5][5];

void insert\_queue(int x)

{

q[++rear]=x;

}

int delete\_queue()

{

return q[++front];

}

int isEmpty\_queue()

{

if(front==rear)

return 1;

else

return 0;

}

void BFS(int v)

{

insert\_queue(v);

while(!isEmpty\_queue())

{

x = delete\_queue();

printf("%d ",x);

visited[v]=1;

for(i=1;i<=n;i++)

{

if(a[v][i]==1 && visited[i]==0)

{

insert\_queue(i);

visited[i]=1;

}

}

}

}

int main()

{

int i,j,v;

printf("Enter no of vertices\n");

scanf("%d",&v);

printf("Enter adjacency matrix\n");

for(i=1;i<=n;i++)

{

for(j=1;j<=n;j++)

{

scanf("%d",&a[i][j]);

}

}

for(i=1;i<=n;i++)

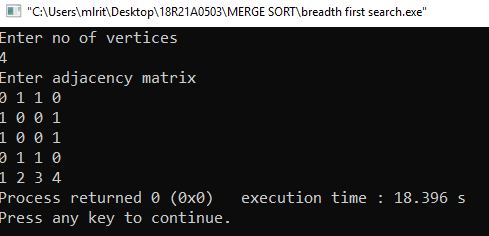
visited[i]=0;

BFS(1);

return 0;

}

**Output:**



**CODE:**

**DEPTH FIRST SEARCH:**

#include <stdio.h>

#define size 5

int s[size],visited[size],i,j,n,top=-1,a[5][5];

void push(int x)

{

s[++top]=x;

}

int isEmpty\_stack()

{

if(top<0)

return 1;

else

return 0;

}

void DFS(int v)

{

int x,flag=0;

push(v);

visited[v]=1;

printf("%d ",v);

while(!isEmpty\_stack())

{

flag=0;

x=s[top];

for(i=1;i<=n;i++)

{

if(a[x][i]==1 && visited[i]==0)

{

push(i);

printf("%d ",i);

visited[i]=1;

flag=1;

break;

}

}

if(flag==0)

top--;

}

}

int main()

{

printf("Enter no of vertices\n");

scanf("%d",&n);

printf("Enter adjacency matrix\n");

for(i=1;i<=n;i++)

{

for(j=1;j<=n;j++)

{

scanf("%d",&a[i][j]);

}

}

for(i=1;i<=n;i++)

visited[i]=0;

DFS(1);

return 0;

}

**Output:**

