

LAB 1

AIM:-To find the dfs and bfs by using adjacent matrix

Code:-

```
#include<stdio.h>
```

```
int q[20],top=-1,front=-1,rear=-1,a[20][20],vis[20],stack[20];
```

```
int delete();
```

```
void add(int item);
```

```
void bfs(int s,int n);
```

```
void dfs(int s,int n);
```

```
void push(int item);
```

```
int pop();
```

```
void main()
```

```
{
```

```
int n,i,s,ch,j;
```

```
char c,dummy;
```

```
printf("ENTER THE NUMBER VERTICES ");
```

```
scanf("%d",&n);
```

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

```
{
```

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

```
{
```

```
printf("ENTER 1 IF %d HAS A NODE WITH %d ELSE 0 ",i,j);
```

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

```
}
```

```
}
```

```
printf("THE ADJACENCY MATRIX IS\n");
```

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

```
{
```

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

```
{
```

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

```
}
```

```
printf("\n");
```

```
}
```

```
do
```

```
{
```

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

```
vis[i]=0;
```

```
printf("\nMENU");
```

```
printf("\n1.B.F.S");
printf("\n2.D.F.S");
printf("\nENTER YOUR CHOICE");
scanf("%d",&ch);
printf("ENTER THE SOURCE VERTEX :");
scanf("%d",&s);
```

```
switch(ch)
{
case 1:bfs(s,n);
break;
case 2:
dfs(s,n);
break;
}
printf("DO U WANT TO CONTINUE(Y/N) ? ");
scanf("%c",&dummy);
scanf("%c",&c);
}while((c=='y')||(c=='Y'));
}
```

```
void bfs(int s,int n)
{
int p,i;
add(s);
vis[s]=1;
p=delete();
if(p!=0)
printf(" %d",p);
while(p!=0)
{
for(i=1;i<=n;i++)
if((a[p][i]!=0)&&(vis[i]==0))
{
add(i);
vis[i]=1;
}
p=delete();
if(p!=0)
printf(" %d ",p);
}
for(i=1;i<=n;i++)
if(vis[i]==0)
bfs(i,n);
}
```

```

void add(int item)
{
if(rear==19)
printf("QUEUE FULL");
else
{
if(rear== -1)
{
q[++rear]=item;
front++;
}
else
q[++rear]=item;
}
}
int delete()
{
int k;
if((front>rear)|| (front== -1))
return(0);
else
{
k=q[front++];
return(k);
}
}

```

```

void dfs(int s,int n)
{
int i,k;
push(s);
vis[s]=1;
k=pop();
if(k!=0)
printf(" %d ",k);
while(k!=0)
{
for(i=1;i<=n;i++)
if((a[k][i]!=0)&&(vis[i]==0))
{
push(i);
vis[i]=1;
}
}
}

```

```
k=pop();
if(k!=0)
printf(" %d ",k);
}
for(i=1;i<=n;i++)
if(vis[i]==0)
dfs(i,n);
}
void push(int item)
{
if(top==19)
printf("Stack overflow ");
else
stack[++top]=item;
}
int pop()
{
int k;
if(top== -1)
return(0);
else
{
k=stack[top--];
return(k);
}
}
```

Out put:-

"C:\Users\Admin\Desktop\1BM21CS059\2ND ADA LAB\LAB2\bin\Debug\LAB2.exe"

```
ENTER THE NUMBER VERTICES 4
ENTER 1 IF 1 HAS A NODE WITH 1 ELSE 0 0
ENTER 1 IF 1 HAS A NODE WITH 2 ELSE 0 1
ENTER 1 IF 1 HAS A NODE WITH 3 ELSE 0 1
ENTER 1 IF 1 HAS A NODE WITH 4 ELSE 0 1
ENTER 1 IF 2 HAS A NODE WITH 1 ELSE 0 0
ENTER 1 IF 2 HAS A NODE WITH 2 ELSE 0 0
ENTER 1 IF 2 HAS A NODE WITH 3 ELSE 0 0
ENTER 1 IF 2 HAS A NODE WITH 4 ELSE 0 1
ENTER 1 IF 3 HAS A NODE WITH 1 ELSE 0 0
ENTER 1 IF 3 HAS A NODE WITH 2 ELSE 0 0
ENTER 1 IF 3 HAS A NODE WITH 3 ELSE 0 0
ENTER 1 IF 3 HAS A NODE WITH 4 ELSE 0 0
ENTER 1 IF 4 HAS A NODE WITH 1 ELSE 0 0
ENTER 1 IF 4 HAS A NODE WITH 2 ELSE 0 0
ENTER 1 IF 4 HAS A NODE WITH 3 ELSE 0 1
ENTER 1 IF 4 HAS A NODE WITH 4 ELSE 0 0
THE ADJACENCY MATRIX IS
0 1 1 1
0 0 0 1
0 0 0 0
0 0 1 0

MENU
1.B.F.S
2.D.F.S
ENTER YOUR CHOICE1
ENTER THE SOURCE VERTEX :1
1 2 3 4 DO U WANT TO CONTINUE(Y/N) ? y

MENU
1.B.F.S
2.D.F.S
ENTER YOUR CHOICE2
ENTER THE SOURCE VERTEX :1
1 4 3 2 DO U WANT TO CONTINUE(Y/N) ?
```

Notes:-

LAB-2

15th June.

Aim:- To find out the DFS and BFS by using adjacent matrix.

Code:-

```
#include <stdio.h>
int q[20], to = -1, front = -1, rear = -1, a[20][20], stack[20];
int deletec;
void add(int item);
void bfs(int s, int u);
void dfs(int s, int u);
void push(int item);
int pop();
void main()
{
    int v, i, s, ch;
    char c;
    printf("Enter the Number vertices");
    scanf("%d", &v);
    for (i = 1; i <= v; i++)
    {
        for (j = 1; j <= v; j++)
        {
            printf("Enter 1 if %d has a Node with %d else 0", i, j);
            scanf("%d", &a[i][j]);
        }
    }
    printf("The Adjacency matrix is\n");
    for (i = 1; i <= v; i++)
    {
        printf("%d", a[i][1]);
    }
    printf("\n");
}
```



```

do {
    for (ci = 1; ci <= n; ci++)
        vis[ci] = 0;

    printf("main menu");
    printf("1. B.F.S");
    printf("2. D.F.S");
    printf("3. Enter your choice");
    scanf("%d", &ch);
    printf("Enter the source vertex:");
    scanf("%d", &s);

    switch(ch)
    {
        case 1: bfs(s, n);
        break;
        case 2:
            dfs(s, n);
            break;
    }
    printf("Do u want to continue (Y/N)?");
    scanf("%c", &dummy);
    scanf("%c", &c);
} while (cc == 'Y' || cc == 'y');

void bfs(int s, int n)
{
    int p, i;
    add(s);
    vis[s] = 1;
    p = delete();
}

```

```

if (cp != 0)
    printf("%d", p);
for (ci = 1; ci <= n; ci++)
    while (cp != 0)
    {
        for (ci = 1; ci <= n; ci++)
            if (adj[ci][s] != 0)
            {
                add(ci);
                vis[ci] = 1;
            }
        p = delete();
        if (cp != 0)
            printf("%d", p);
    }
    for (ci = 1; ci <= n; ci++)
        if (vis[ci] == 0)
            bfs(ci, n);
}

void add(int it)
{
    if (rear == 10)
        printf("Queue full");
    else
    {
        if (rear == -1)
        {
            q[rear] = it;
            front++;
        }
        else
            q[rear] = it;
    }
}

```



```
if (cp != 0)
    printf("%d", p);
```

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

```
while (cp != 0)
```

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

```
if (arr[i] != 0 && vis[i] == 0)
```

```
{
    add(i);
```

```
vis[i] = 1;
```

```
p = delete();
```

```
if (cp != 0)
    printf("%d", p);
```

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

```
if (vis[i] == 0)
```

```
    brr[i] = 1;
```

```
void add(int item)
```

```
{
    if (rear == -1)
```

```
        printf("Queue full");
```

```
    else
```

```
    {
        if (rear == -1)
```

```
            arr[rear] = item;
```

```
            front++;
```

```
    }
    else
```

```
        arr[rear] = item;
```



```

}
}
int delete()
{
    int k;
    if (cf == front > rear) || cf == -1)
    {
        return 0;
    }
    else
    {
        k = q[cf];
        return k;
    }
}

void dfs(int s, int u)
{
    int i, k;
    push(s);
    vis[s] = 1;
    k = pop();
    if (k != 0)
    {
        printf("%d ", k);
        while (k != 0)
        {
            for (ci = 1; ci <= n; ci++)
            {
                if (adj[k][ci] == 0 && vis[ci] == 0)
                {
                    push(ci);
                    vis[ci] = 1;
                }
            }
            k = pop();
        }
        if (k != 0)
        {
            printf("%d ", k);
        }
    }
}

```

```

for (ci = 1; ci <= n; ci++)
{
    if (vis[ci] == 0)
    {
        dfs(ci, u);
    }
}

void push(int i)
{
    if (top == -1)
    {
        printf("Stack is empty\n");
    }
    else
    {
        stack[++top] = i;
    }
}

int pop()
{
    if (top == -1)
    {
        return 0;
    }
    else
    {
        k = stack[top--];
        return k;
    }
}

```

Output :-

Enter the number of nodes

Enter 1 if 1 is connected to 2

Enter 1 if 2 is connected to 1

Enter 1 if 1 is connected to 3

Enter 1 if 3 is connected to 1

The Adjacency List is

```

0 1 1
0 0 0
0 0 0
0 0 1

```



```

for (i=1; i<=n; i++)
    if (vis[i]==0)
        dfs(i, n);
}

void push(int item)
{
    if (top == 19)
        printf("Stack overflow");
    else
        stack[top++] = item;
}

int pop()
{
    if (top < 0)
        return 0;
    else
        return stack[top--];
}

```

Output :-

Enter the Number of vertices 4
 Enter 1 if 1 has a node with 1 else 0 0
 Enter 1 if 2 has a node with 1 else 0 0
 Enter 1 if 3 has a node with 1 else 0 0
 Enter 1 if 4 has a node with 1 else 0 0

The ADJACENCY MATRIX IS

```

0 1 1 1
0 0 0 1
0 0 0 0
0 0 1 0

```


MENU

1. B.F.S

2. D.F.S

Enter your choice:

Enter the source vertex:

1 2 3 4 Do you want to continue (Y/N)

Menu

1. B.F.S

2. D.F.S

Enter your choice:

Enter the source vertex:

1 2 3 4 Do you want to continue (Y/N)

200
10/10