```c
#include<stdio.h>

int q[20],top=-1,front=0,rear=-1,adj[20][20],v[20],stack[20],n;
int dequeue();
void enqueue(int value);
void bfs();
void dfs();
void push(int value);
int pop();

void main()
{
int i,k,m,j;
printf("Enter the no of nodes ");
scanf("%d",&n);

for(i=1;i<=n;i++)
{
 for(j=1;j<=n;j++)
 adj[i][j]=0;
}

for(i=1;i<=n;i++)
{printf("Enter the no of neighbouring nodes of %d ",i);
scanf("%d",&m);
for(j=1;j<=m;j++)
{printf("Enter the neighbouring node of %d ",i);
    scanf("%d",&k);
    adj[i][k]=1;
}
}
printf("THE ADJACENCY MATRIX IS\n");
for(i=1;i<=n;i++)
{
for(j=1;j<=n;j++)
{
printf(" %d",adj[i][j]);
}
printf("\n");
}
printf("BFS:-");
bfs();
printf("DFS:-");
dfs();
}



//**************BFS(breadth-first search) code**************//
void bfs()
{
    int source,a,i;
for(i=1;i<=n;i++)
v[i]=0;

printf("Enter the source node");
```

```
scanf("%d",&source);
enqueue(source);
v[source]=1;
a=dequeue();
while(a!=-1)
{printf("%d ",a);
 for(i=1;i<=n;i++)
 {if(adj[a][i]==1&&v[i]==0)
  {enqueue(i);
   v[i]=1;
  }
 }
 a=dequeue();
}
}


void enqueue(int value)
{rear++;
q[rear]=value;
}

int dequeue()
{
int t;
if(front>rear)
return(-1);
else
{
t=q[front];front++;
return(t);
}
}

//**************DFS(depth-first search) code**************//
void dfs()
{
    int source,a,i;
for(i=1;i<=n;i++)
v[i]=0;

printf("Enter the source node");
scanf("%d",&source);
push(source);
v[source]=1;
a=pop();
while(a!=-1)
{printf("%d ",a);
 for(i=1;i<=n;i++)
 {if(adj[a][i]==1&&v[i]==0)
  {push(i);
   v[i]=1;
  }
 }
 a=pop();
}
}


void push(int value)
{top++;
q[top]=value;
}

int pop()
{
int t;
if(top==-1)
return(-1);
else
{
t=q[top];top--;
return(t);
}
}
```