LAB PROGRAM 9:

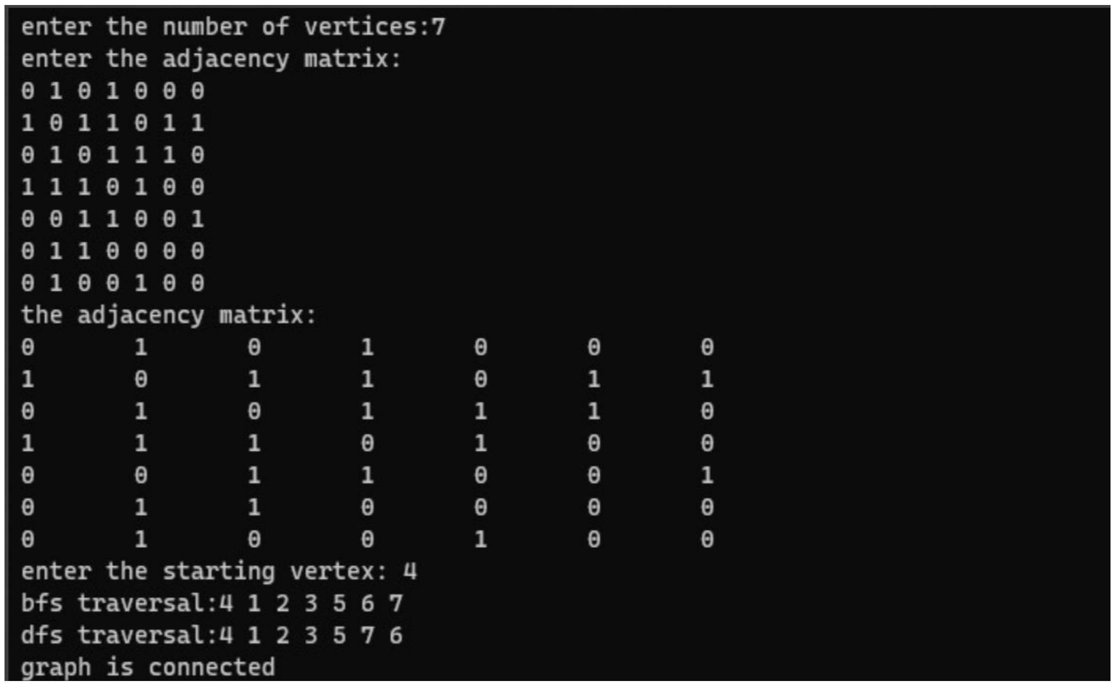
Write a program to traverse a graph using BFS method.
Write a program to check whether given graph is connected or not using

DFS method CODE:

```
#include <stdio.h> #include
<stdlib.h>
struct node{     int
data;     struct node
*next;
}*front=NULL,*rear=NULL; void
enquque(int x){
   struct node t=(struct node) malloc(sizeof(struct node));
if(t==NULL){
      prin]("queue is overflow");
   }
   else{           t-
>data=x;           t-
>next=NULL;
if(front==NULL){
front=rear=t;


   }
   else{       rear-
>next=t;
      rear=t;
   }
   }
}
int dequque(){     struct node
*t;     int x=-1;
if(front==NULL){
prin]("queque is empty");
return x;
   }
   else{            t=front;
x=t->data;
front=front->next;
free(t);     return x;

   }
}
int isempt(){
if(front==NULL){
       return 1;
   }
   return 0;
}
//traverse a graph using BFS void bfs(int
i,int visited[],int a[][20],int n){
   int u;     prin]("bfs traversal:");
prin]("%d ",i);     visited[i-1]=1;
enquque(i-1);     while(!isempt()){
u=dequque();            for(int
v=0;v<n;v++){                if(a[u][v]==1 &&
visited[v]==0){                   prin]("%d
",v+1);                  visited[v]=1;
                 enquque(v);
              }
}

   }
}
//connected or not using DFS void dfs(int
i,int visited[],int a[][20],int n){
if(visited[i-1]==0){
       prin]("%d ",i);        visited[i-1]=1;
for(int j=0;j<n;j++){            if(a[i-
1][j]==1 && visited[j]==0){
              dfs(j+1,visited,a,n);
           }
       }
   }
}
void main(){     int
visited[20]={0};     int
a[20][20];     int
n,first;     int
count=0;
   prin]("enter the number of ver,ces:");
scanf("%d",&n);     prin]("enter the
adjacency matrix:");
   for(int i=0;i<n;i++){
for(int j=0;j<n;j++){
         scanf("%d",&a[i][j]);
      }
   }
   prin]("the adjacency matrix:\n");
for(int i=0;i<n;i++){        for(int
j=0;j<n;j++){
         prin]("%d\t",a[i][j]);
      }
      prin]("\n");
   }
   prin]("enter the star,ng vertex: ");
scanf("%d",&first);
bfs(first,visited,a,n);     for(int
i=0;i<20;i++){
      visited[i]=0;
   }
   prin]("\ndfs traversal:");
dfs(first,visited,a,n);     for(int
i=0;i<n;i++){
if(visited[i]==1){
         count++;
      }
   }
   if(count==n){
      prin]("\ngraph is connected");
```

```
}
else{
   prin]("\ngraph is not connected");
} }
```

OUTPUT:



```
enter the number of vertices:7
enter the adjacency matrix:
0 1 0 1 0 0 0
1 0 1 1 0 1 1
0 1 0 1 1 1 0
1 1 1 0 1 0 0
0 0 1 1 0 0 1
0 1 1 0 0 0 0
0 1 0 0 1 0 0
the adjacency matrix:
0       1       0       1       0       0       0
1       0       1       1       0       1       1
0       1       0       1       1       1       0
1       1       1       0       1       0       0
0       0       1       1       0       0       1
0       1       1       0       0       0       0
0       1       0       0       1       0       0
enter the starting vertex: 4
bfs traversal:4 1 2 3 5 6 7
dfs traversal:4 1 2 3 5 7 6
graph is connected
```