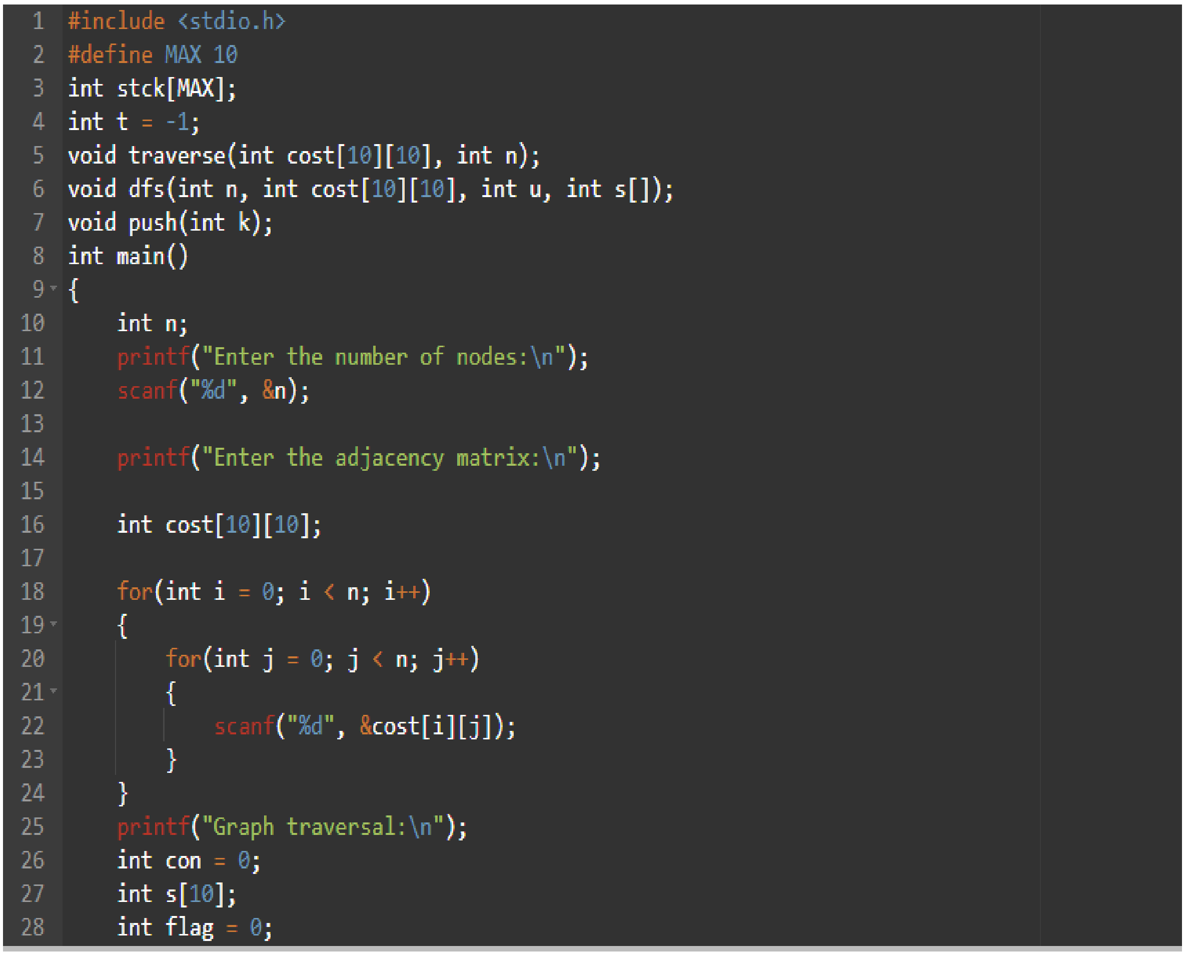
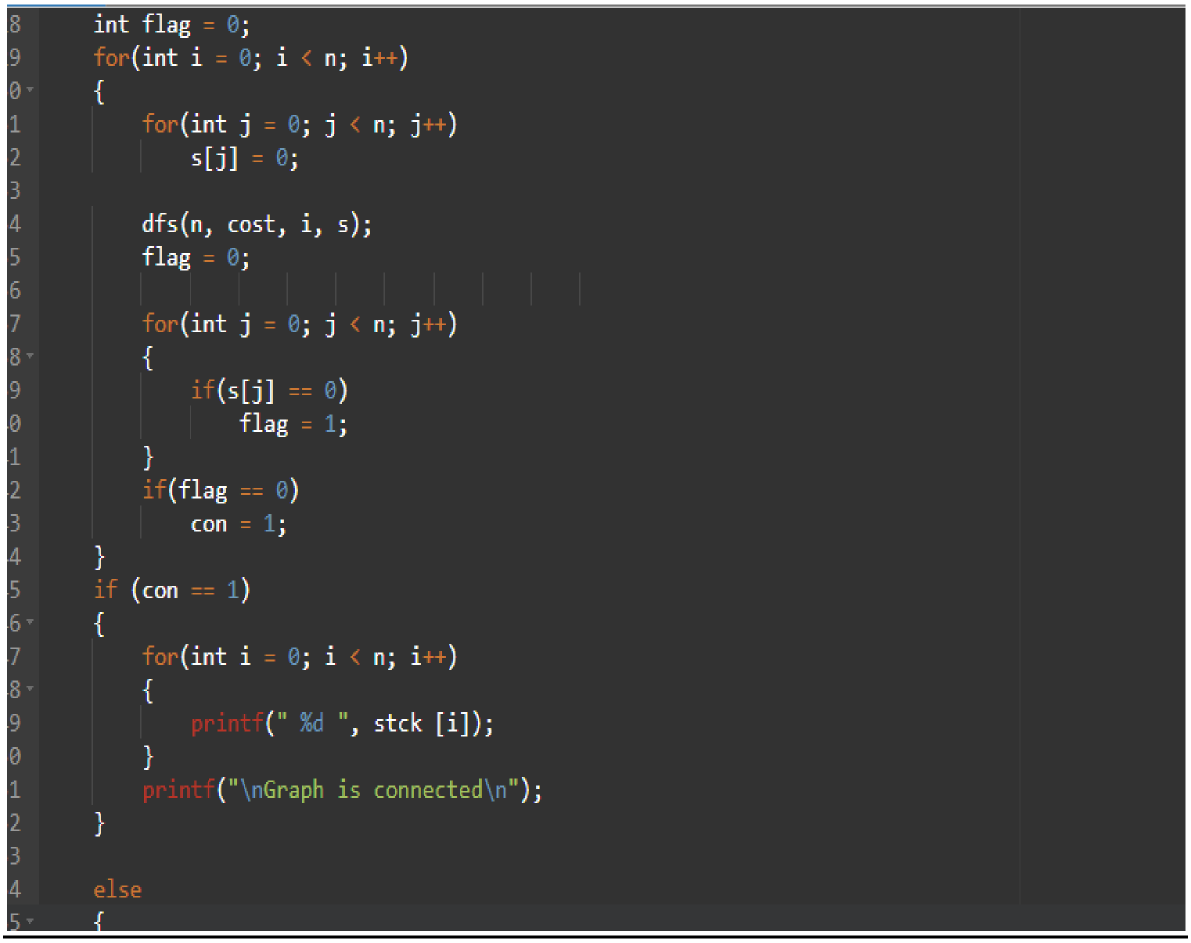
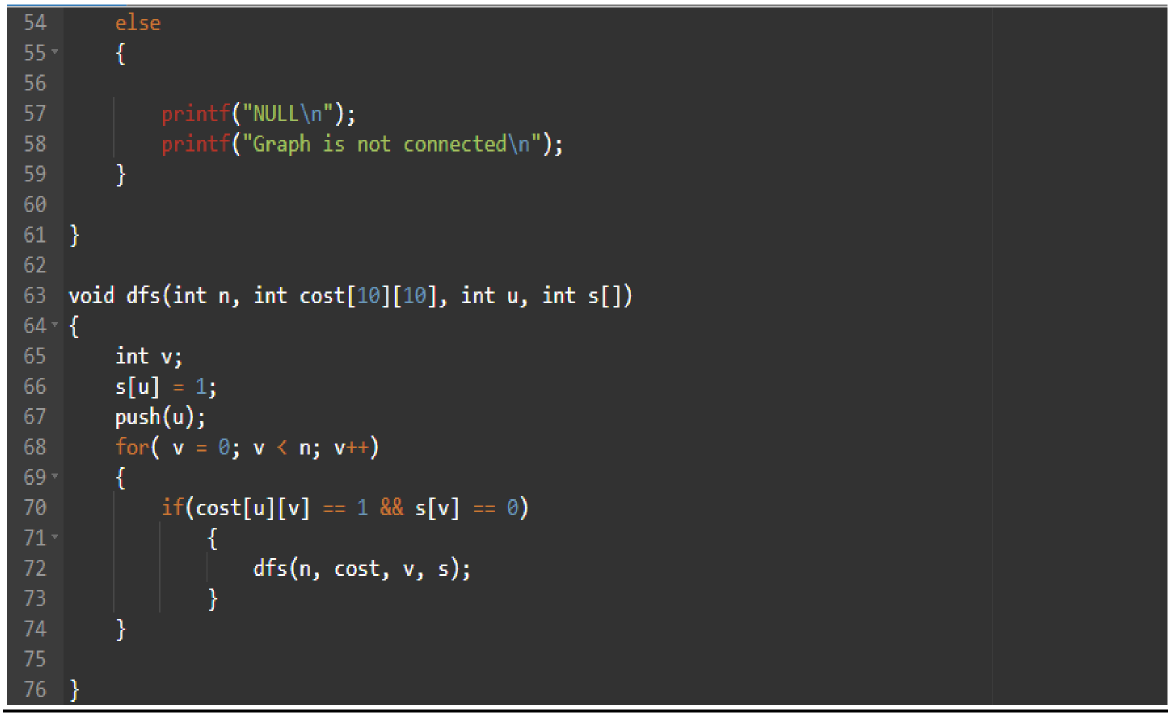
Dfs:



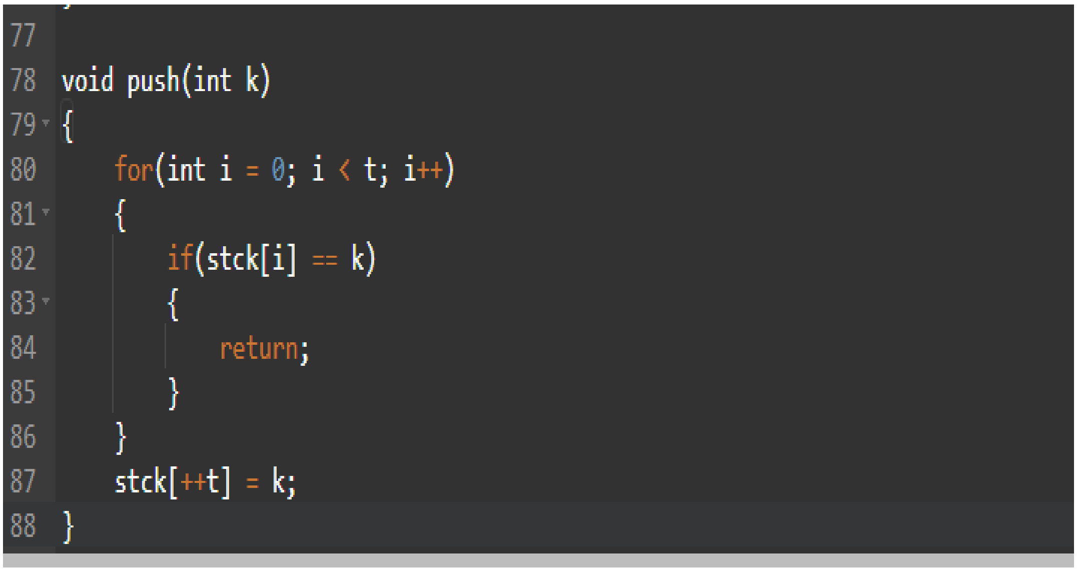
```c
#include <stdio.h>
#define MAX 10
int stck[MAX];
int t = -1;
void traverse(int cost[10][10], int n);
void dfs(int n, int cost[10][10], int u, int s[]);
void push(int k);
int main()
{
    int n;
    printf("Enter the number of nodes:\n");
    scanf("%d", &n);

    printf("Enter the adjacency matrix:\n");

    int cost[10][10];

    for(int i = 0; i < n; i++)
    {
        for(int j = 0; j < n; j++)
        {
            scanf("%d", &cost[i][j]);
        }
    }
    printf("Graph traversal:\n");
    int con = 0;
    int s[10];
    int flag = 0;
```



```
    int flag = 0;
    for(int i = 0; i < n; i++)
    {
        for(int j = 0; j < n; j++)
            s[j] = 0;

        dfs(n, cost, i, s);
        flag = 0;

        for(int j = 0; j < n; j++)
        {
            if(s[j] == 0)
                flag = 1;
        }
        if(flag == 0)
            con = 1;
    }
    if (con == 1)
    {
        for(int i = 0; i < n; i++)
        {
            printf(" %d ", stck [i]);
        }
        printf("\nGraph is connected\n");
    }

    else
    {
```

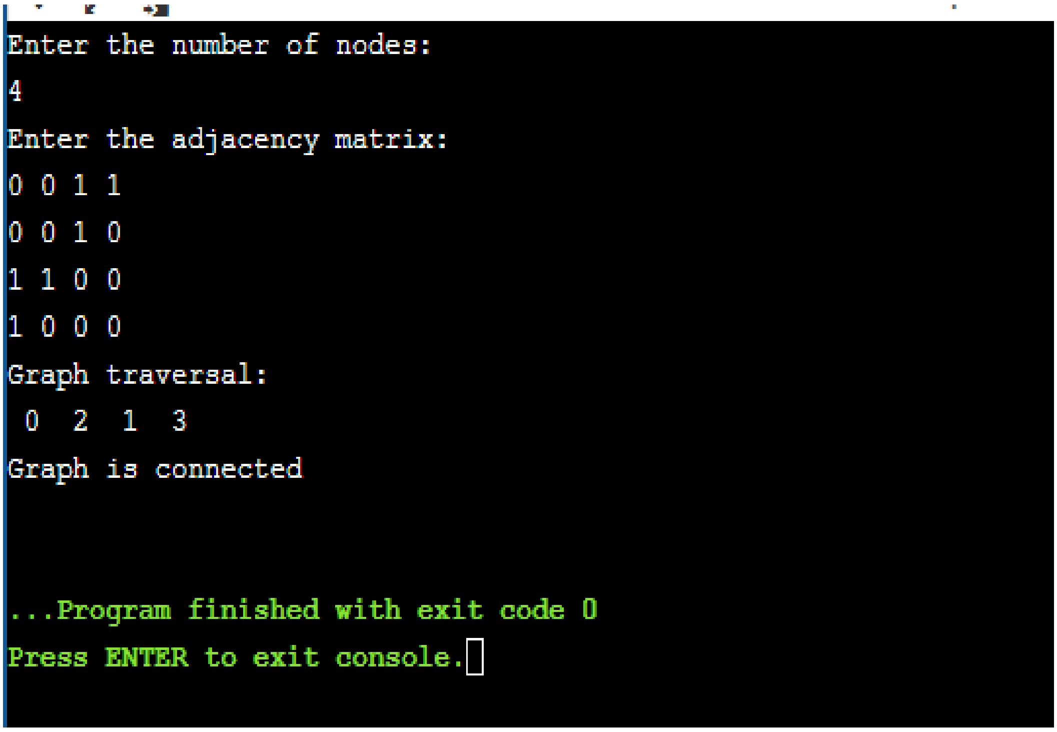


```
    else
    {

        printf("NULL\n");
        printf("Graph is not connected\n");
    }

}

void dfs(int n, int cost[10][10], int u, int s[])
{
    int v;
    s[u] = 1;
    push(u);
    for( v = 0; v < n; v++)
    {
        if(cost[u][v] == 1 && s[v] == 0)
            {
                dfs(n, cost, v, s);
            }
    }

}
```



```
}

void push(int k)
{
    for(int i = 0; i < t; i++)
    {
        if(stck[i] == k)
        {
            return;
        }
    }
    stck[++t] = k;
}
```

Output:



```
Enter the number of nodes:
4
Enter the adjacency matrix:
0 0 1 1
0 0 1 0
1 1 0 0
1 0 0 0
Graph traversal:
 0  2  1  3
Graph is connected


...Program finished with exit code 0
Press ENTER to exit console.
```

```
#include<stdio.h>
void tower_hanoi(int n, char src, char temp, char dest) {
 if(n == 1) {
               printf("Move %d disc from %c to %c \n",n,src,dest);
               return ;
   }
```

```
            tower_hanoi(n - 1, src, dest, temp);


            printf("Move %d disc from %c to %c \n",n,src,dest);


            tower_hanoi(n - 1, temp, src, dest);
}


int main()
{
   int x;


            printf("Enter no of disc");
   scanf("%d",&x);


            tower_hanoi(x, 'A', 'B', 'C');


   return 0;
}
```

#include #include int a[1][10]; void dfs(int n, int cost[10][10], int u, int s[]) { int v; s[u]=1; for(v=0;v

```
#include<stdio.h>
#include<conio.h>
int a[1][10];
void dfs(int n, int cost[10][10], int u, int s[])
{
```

```
int v;

s[u]=1;

for(v=0;v<n;v++)

{

if((cost[u][v]==1) && (s[v]==0))

dfs(n,cost,v,s);

}

}

void main()

{

int n,i,j,cost[10][10],s[10],con,flag;

//clrscr();

printf("Enter the number of nodes\n");

scanf("%d", &n);

printf("Enter the adjacency matrix\n");

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

{

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

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

}

con=0;

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

{

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

s[i]=0;

dfs(n,cost,j,s);

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

{

if(s[i]==0)

flag=1;

}

if(flag==0)

con=1;

}

if(con==1)

printf("Graph is connected\n");

else

printf("Graph is not connected\n");

getch();

}
```