connected_dfs.c | nodes_dfs.c | pattern_bruteforce.c

nodes_dfs.c

```c
/*PROGRAM TO PRINT ALL THE NODES REACHABLE FROM A GIVEN STARTING NODE IN A GIVEN DIRECTED GRAPH USING DFS METHOD*/
#include<stdio.h>
#include<conio.h>
void dfs(int);
int arr[20][20],visit[20],n;
void main()
{
    int i,j,source;
    printf("\nEnter the number of vertices:");
    scanf("%d",&n);
    printf("Enter adjacency matrix:\n");
    for(i=1;i<=n;i++)
    {
        for(j=1;j<=n;j++)
        {
            scanf("%d",&arr[i][j]);
        }
    }
    for(i=1;i<=n;i++)
    {
        visit[i]=0;
    }
    printf("\nEnter source vertex:");
    scanf("%d",&source);
    printf("Nodes reachable from source vertex %d:\n",source);
    dfs(source);
    getch();
}
void dfs(int v)
{
    int i;
    visit[v]=1;
    printf("%d ",v);
    for(i=1;i<=n;i++)
    {
        if(arr[v][i]==1 && visit[i]==0)
        {
            dfs(i);
```

connected_dfs.c | nodes_dfs.c | pattern_bruteforce.c

nodes_dfs.c

```c
        }
    }
    for(i=1;i<=n;i++)
    {
        visit[i]=0;
    }
    printf("\nEnter source vertex:");
    scanf("%d",&source);
    printf("Nodes reachable from source vertex %d:\n",source);
    dfs(source);
    getch();
}
void dfs(int v)
{
    int i;
    visit[v]=1;
    printf("%d ",v);
    for(i=1;i<=n;i++)
    {
        if(arr[v][i]==1 && visit[i]==0)
        {
            dfs(i);
        }
    }
}

```

connected_dfs.c | nodes_dfs.c | pattern_bruteforce.c

nodes_dfs.c

```
{
    int i,j,source;
    printf("\nEnter the number of vertices:");
    scanf("%d",&n);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: powershell

```
PS C:\Users\muska\OneDrive\Desktop\C programs> gcc nodes_dfs.c
PS C:\Users\muska\OneDrive\Desktop\C programs> .\a.exe

Enter the number of vertices:10
Enter adjacency matrix:
0 1 1 1 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 0 0 0
0 0 0 0 0 0 0 1 0 0
0 0 1 0 0 0 0 1 0 0
0 0 0 0 0 0 0 1 1 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 1 0

Enter source vertex:1
Nodes reachable from source vertex 1:
1 2 5 8 3 4 6 7 9
PS C:\Users\muska\OneDrive\Desktop\C programs> gcc pattern_bruteforce.c
PS C:\Users\muska\OneDrive\Desktop\C programs> .\a.exe
```

connected_dfs.c | nodes_dfs.c | pattern_bruteforce.c

connected_dfs.c

```c
/*PROGRAM TO CHECK WHETHER GIVEN GRAPH IS CONNECTED OR NOT USING DFS METHOD */
#include<stdio.h>
#include<conio.h>
void dfs(int);
int arr[20][20],visit[20],n;
void main()
{
    int i,j,source,count=0;
    printf("\nEnter the number of vertices:");
    scanf("%d",&n);
    printf("Enter adjacency matrix:\n");
    for(i=1;i<=n;i++)
    {
        for(j=1;j<=n;j++)
        {
            scanf("%d",&arr[i][j]);
        }
    }
    for(i=1;i<=n;i++)
    {
        visit[i]=0;
    }
    printf("\nEnter source vertex:");
    scanf("%d",&source);
    printf("Nodes reachable from source vertex %d:\n",source);
    dfs(source);
    for(i=1;i<=n;i++)
    {
        if(visit[i]==1)
        {
            count++;
        }
    }
    if(count==n)
    {
        printf("\nGraph is connected");
    }
    else
```

connected_dfs.c | nodes_dfs.c | pattern_bruteforce.c

connected_dfs.c

```c
    scanf("%d",&source);
    printf("Nodes reachable from source vertex %d:\n",source);
    dfs(source);
    for(i=1;i<=n;i++)
    {
        if(visit[i]==1)
        {
            count++;
        }
    }
    if(count==n)
    {
        printf("\nGraph is connected");
    }
    else
    {
        printf("\nGraph is not connected");
    }
    getch();
}
void dfs(int v)
{
    int i;
    visit[v]=1;
    printf("%d ",v);
    for(i=1;i<=n;i++)
    {
        if(arr[v][i]==1 && visit[i]==0)
        {
            dfs(i);
        }
    }
}
```

connected_dfs.c | nodes_dfs.c | pattern_bruteforce.c

connected_dfs.c

```
47      visit[v]=1;
48      printf("%d ",v);
49      for(i=1;i<=n;i++)
50      {
51          if(arr[v][i]==1 && visit[i]==0)
52          {
53              dfs(i);
54          }
55      }
56  }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: powershell

```
Graph is not connected
PS C:\Users\muska\OneDrive\Desktop\C programs> gcc connected_dfs.c
PS C:\Users\muska\OneDrive\Desktop\C programs> .\a.exe

Enter the number of vertices:5
Enter adjacency matrix:
0 0 0 1 1
0 0 0 1 1
0 0 0 0 0
1 1 0 0 0
1 1 0 0 0

Enter source vertex:1
Nodes reachable from source vertex 1:
1 4 2 5
Graph is not connected
PS C:\Users\muska\OneDrive\Desktop\C programs> gcc nodes_dfs.c
PS C:\Users\muska\OneDrive\Desktop\C programs> .\a.exe
```