

9 b) Write a program to check whether given graph is connected or not using DFS method.

```
#include <stdio.h>

#define MAX 10

int visited[MAX];

/* DFS function */
void DFS(int graph[MAX][MAX], int n, int v)
{
    int i;
    visited[v] = 1;

    for (i = 0; i < n; i++)
    {
        if (graph[v][i] == 1 && !visited[i])
        {
            DFS(graph, n, i);
        }
    }
}

int main()
{
    int graph[MAX][MAX], n, i, j;

    printf("Enter number of vertices: ");
    scanf("%d", &n);

    printf("Enter adjacency matrix:\n");
    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
            scanf("%d", &graph[i][j]);

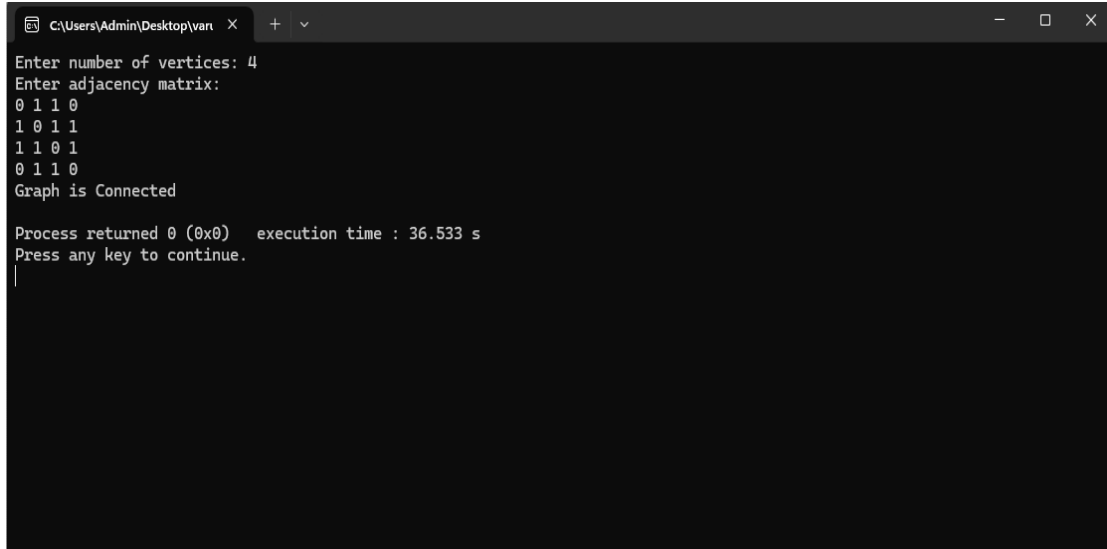
    /* Start DFS from vertex 0 */
    DFS(graph, n, 0);

    /* Check if all vertices are visited */
    for (i = 0; i < n; i++)
    {
        if (!visited[i])
        {
            printf("Graph is NOT Connected\n");
            return 0;
        }
    }
}
```

```
}

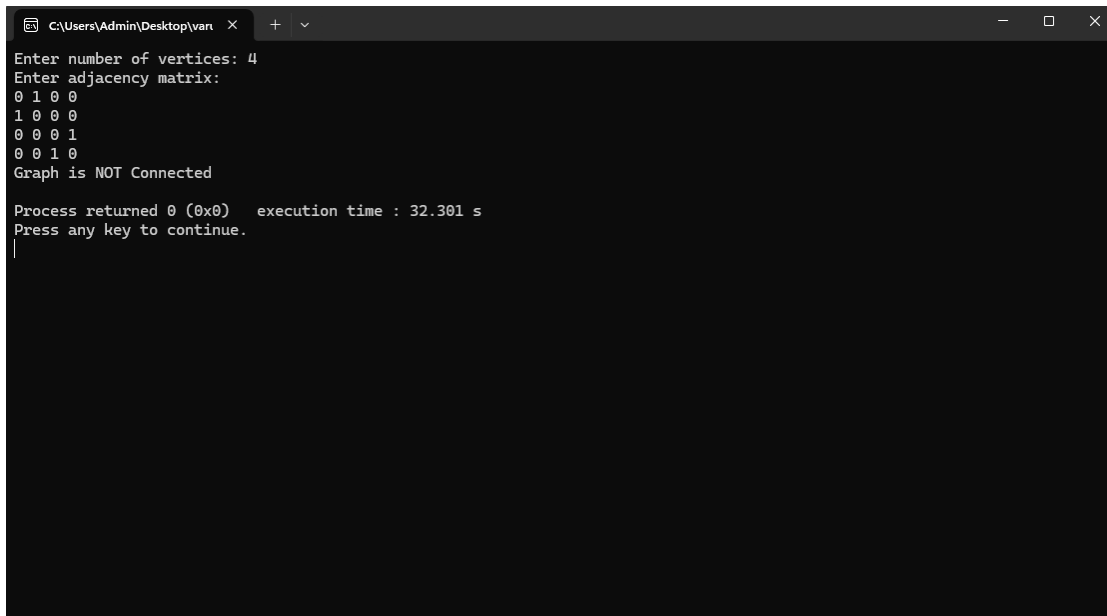
printf("Graph is Connected\n");
return 0;
}
```

Output:



```
C:\Users\Admin\Desktop\var  X  +  v  -  □  X
Enter number of vertices: 4
Enter adjacency matrix:
0 1 1 0
1 0 1 1
1 1 0 1
0 1 1 0
Graph is Connected

Process returned 0 (0x0)   execution time : 36.533 s
Press any key to continue.
|
```



```
C:\Users\Admin\Desktop\var  X  +  v  -  □  X
Enter number of vertices: 4
Enter adjacency matrix:
0 1 0 0
1 0 0 0
0 0 0 1
0 0 1 0
Graph is NOT Connected

Process returned 0 (0x0)   execution time : 32.301 s
Press any key to continue.
|
```