Green University

Green University of Bangladesh

Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: 3rd - Fall semester / Year:2022

BSc in CSE (Day)

LAB REPORT NO - 06

Course Title: Data Structure Lab

Course Code: CSE 106 / Section: DE-221

Lab Experiment Tittle :

Implement a C program for graph traversal using DFS

Student Details

	Name	ID
1.	Khondokar Saim	221902353

Lab Date	: 14 / 12 / 2022
Submission Date	: 21 / 12 / 2022
Course Teacher's Name	: Farhana Akter Sunny.

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: ....   Signature: ....

Comments: ....   Date: ....

❑ Tittle of the Lab Experiment :

Implement a C program for graph traversal using DFS.

❑ Objectives :

- Traversing a tree or graph: DFS can be used to traverse all the nodes in a tree or graph, starting from the root node or any other starting node.
- Searching for a specific node or value: DFS can be used to search for a specific node or value in a tree or graph.
- Checking if a path exists between two nodes: DFS can be used to check if there is a path between two nodes in a tree or graph.
- Finding connected components in a graph: DFS can be used to find the connected components in an undirected graph, i.e., the subgraphs in which all nodes are connected to each other.
- Topological sorting: DFS can be used to topologically sort the nodes in a directed acyclic graph (DAG). A topological sort is an ordering of the nodes such that for every edge u -> v, the node u appears before the node v in the ordering.
- Solving puzzles or problems: DFS can be used to solve puzzles or problems that involve searching through a large number of possibilities, such as the Tower of Hanoi puzzle or the n-queens problem.

1. **Implement a C program for graph traversal using DFS.**

Algorithm :

Step – 1 = Start at the root node or any other starting node.

Step – 2 = Explore each branch as far as possible until you reach a leaf node (a node with no children).

Step – 3 = If the current node has no children, backtrack to the most recent node that has unexplored children and explore the next child.

Step – 4 = Repeat this process until you have visited all the nodes in the tree.

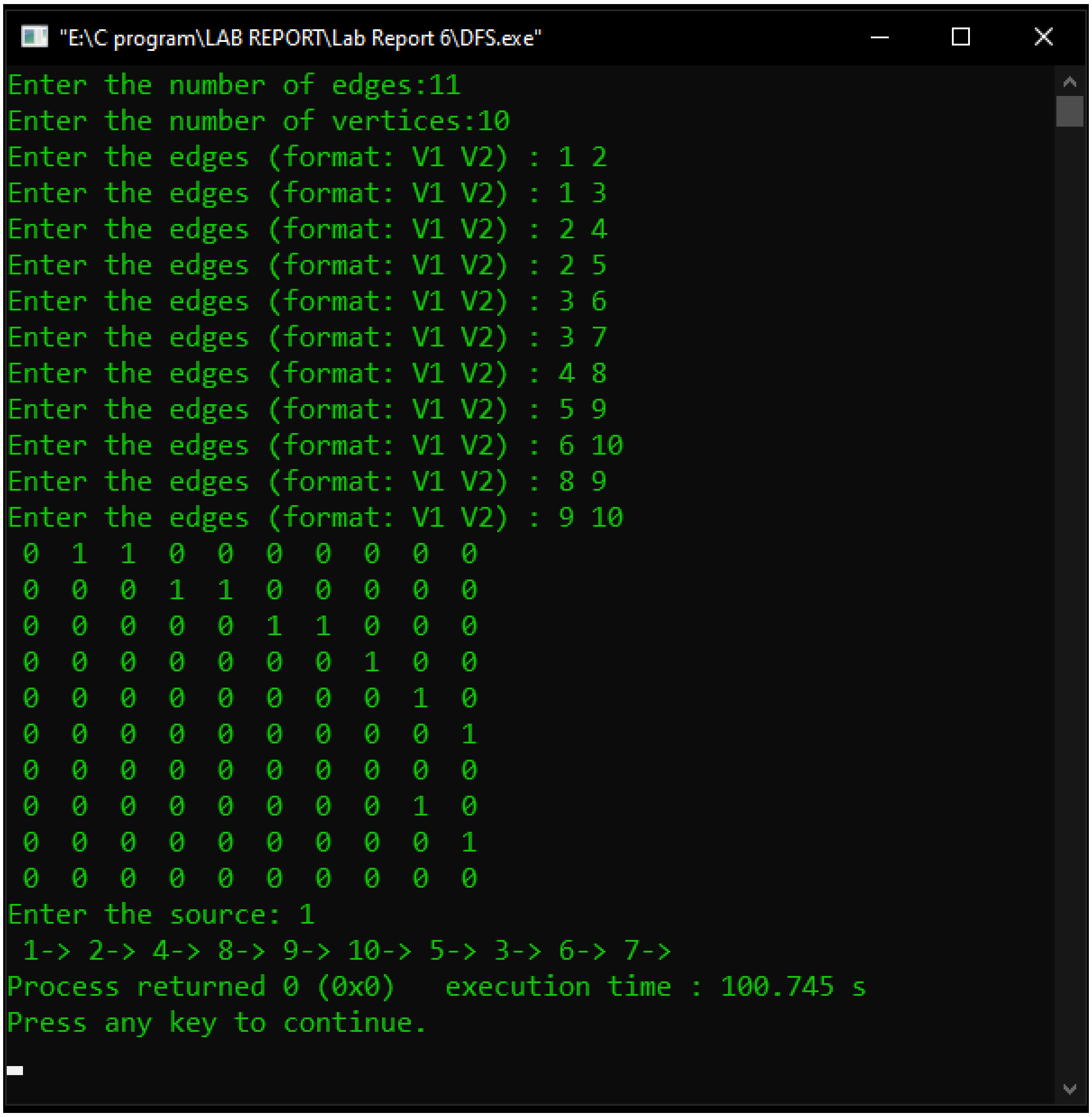
Step – 5 = Here is some pseudocode that demonstrates the DFS algorithm:

```
DFS(node):
  // Mark the node as visited
  mark(node)
  // Explore each child of the node
  for each child in children(node):
    if child is not visited:
      DFS(child)
```

Code:

```
#include <stdio.h>
#include <stdlib.h>

int source,V,E,time,visited[20],G[20][20];
void DFS(int i)
{
    int j;
    visited[i]=1;
    printf(" %d->",i+1);
    for(j=0;j<V;j++)
    {
        if(G[i][j]==1&&visited[j]==0)
            DFS(j);
    }
}
int main()
{
    int i,j,v1,v2;

    printf("Enter the number of edges:");
    scanf("%d",&E);
    printf("Enter the number of vertices:");
    scanf("%d",&V);
    for(i=0;i<V;i++)
    {
        for(j=0;j<V;j++)
            G[i][j]=0;
    }

    for(i=0;i<E;i++)
    {
        printf("Enter the edges (format: V1 V2) : ");
        scanf("%d%d",&v1,&v2);
        G[v1-1][v2-1]=1;

    }

    for(i=0;i<V;i++)
    {
        for(j=0;j<V;j++)
            printf(" %d ",G[i][j]);
        printf("\n");
    }
printf("Enter the source: ");
    scanf("%d",&source);
        DFS(source-1);
    return 0;
}
```

Output:



```
"E:\C program\LAB REPORT\Lab Report 6\DFS.exe"
Enter the number of edges:11
Enter the number of vertices:10
Enter the edges (format: V1 V2) : 1 2
Enter the edges (format: V1 V2) : 1 3
Enter the edges (format: V1 V2) : 2 4
Enter the edges (format: V1 V2) : 2 5
Enter the edges (format: V1 V2) : 3 6
Enter the edges (format: V1 V2) : 3 7
Enter the edges (format: V1 V2) : 4 8
Enter the edges (format: V1 V2) : 5 9
Enter the edges (format: V1 V2) : 6 10
Enter the edges (format: V1 V2) : 8 9
Enter the edges (format: V1 V2) : 9 10
 0  1  1  0  0  0  0  0  0  0
 0  0  0  1  1  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  0  0  0  0  0  0  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  1  0
 0  0  0  0  0  0  0  0  0  1
 0  0  0  0  0  0  0  0  0  0
Enter the source: 1
 1-> 2-> 4-> 8-> 9-> 10-> 5-> 3-> 6-> 7->
Process returned 0 (0x0)   execution time : 100.745 s
Press any key to continue.
```

Case – 1 Graph Traversal using DFS

❑ Analysis and Discussion :

- We got the exact result on output. Sometimes the result was wrong but we found the right implementation.
- The problem of displaying anything in output is the easiest implementation. We solve that very easily.
- In this assignment, we faced some problems in this question but with the teacher's help we solve it.
- All program is easy to understand and these helped me a lot to remove my confusion about DFS programming and Graph traversing operations.
- I learnt display something in program, find DFS and graph traversing operations , switch statement and application of user-defined function etc on program and many basic things about c programming.