LAB Manual

PART A

(PART A : TO BE REFFERED BY STUDENTS)

**Experiment No.08**

**A.1 Aim:**

**To implement the Graph traversal techniques to solve the problem.**

**A.2 Prerequisite:**

Prior knowledge of introduction to data structure.

**A.3 Outcome:**

**After successful completion of this experiment students will be able to:**

After successful completion of this experiment students will be able to:

1. Understand the concept of data structures
2. Know the use and implementation of different types Graphs

**A.4 Theory:**

Data structure is logical or mathematical organization of data; it describes how to store the data and access data from memory. Actually in our programming data stored in main memory(RAM) and to develop efficient software or firmware we need to care about memory. To efficiently manage we required data structure.

**GRAPHS:**

A graph is a pictorial representation of a set of objects where some pairs of objects are connected by links. The interconnected objects are represented by points termed as vertices, and the links that connect the vertices are called edges.

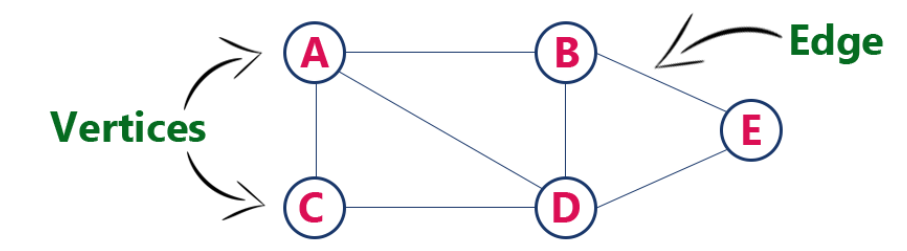
Formally, a graph is a pair of sets (V, E), where V is the set of vertices and E is the set of edges, connecting the pairs of vertices.

**Example**

The following is a graph with 5 vertices and 6 edges.

This graph G can be defined as G = ( V , E )

Where V = {A,B,C,D,E} and E = {(A,B),(A,C)(A,D),(B,D),(C,D),(B,E),(E,D)}.

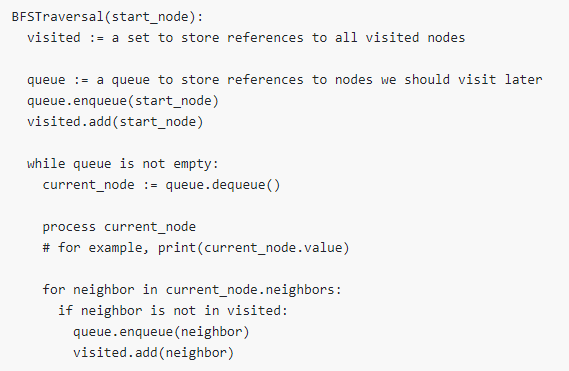


1. **BFS**

**The algorithm works as follows:**

1. Start by putting any one of the graph's vertices at the back of a queue.
2. Take the front item of the queue and add it to the visited list.
3. Create a list of that vertex's adjacent nodes. Add the ones which aren't in the visited list to the back of the queue.
4. Keep repeating steps 2 and 3 until the queue is empty.

**Pseudocode**

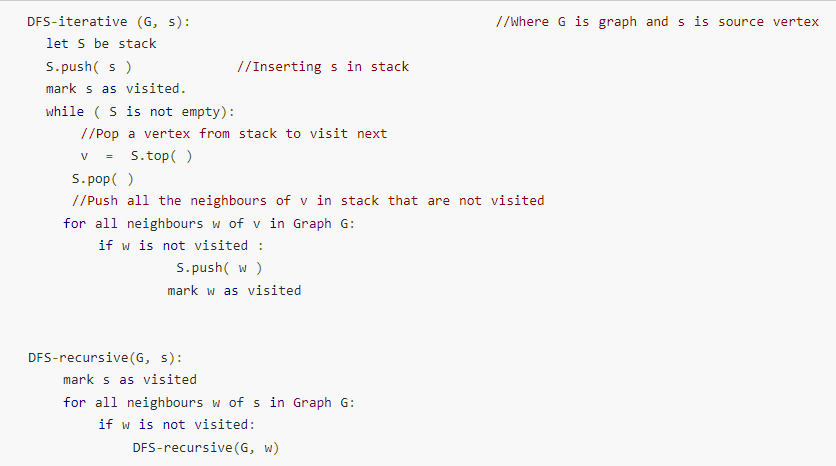


1. **DFS**

**The DFS algorithm works as follows:**

1. Start by putting any one of the graph's vertices on top of a stack.
2. Take the top item of the stack and add it to the visited list.
3. Create a list of that vertex's adjacent nodes. Add the ones which aren't in the visited list to the top of the stack.
4. Keep repeating steps 2 and 3 until the stack is empty.

**Pseudocode**



**A.5 Procedure/Algorithm:**

**A.5.1 TASK 1: Write programs for the following questions:**

**Q1**. Write a program in C/C++ to implement following on a Graph:

1. Breadth First traversal
2. Depth First traversal

Save and close the file and name it as **DS\_Experiment08\_YourName.**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

***(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case there is no Black board access available)***

|  |  |
| --- | --- |
| Roll No. | Name: |
| Class : | Batch : |
| Date of Experiment: | Date of Submission |
| Grade : |  |

**B.1 Answers of Task to be written by student:**

***B.1.1 Code***

***B.1.2 Output***

**B.2 Observations and learning:**

***(Students are expected to comment on the output obtained with clear observations and learning for each task/ sub part assigned)***

**B.3 Conclusion:**

*(****Students must write the conclusion as per the attainment of individual outcome listed above and learning/observation noted in section B.2)***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***