LAB 5

CLO	**0**	**1**	**2**	**3**
CLO2	0 for no problem solved	3.0 Marks for Question 1	3.0 Marks for Question 1	.0 Marks for Question 1

Breadth First Search:

Breadth-first search (BFS) is an algorithm for traversing or searching tree or graph data structures. It starts at the tree root and explores all of the neighbor nodes at the present depth prior to moving on to the nodes at the next depth level. It uses a queue (First In First Out) instead of a stack and it checks whether a vertex has been discovered before enqueueing the vertex. BFS is a traversing algorithm where you should start traversing from a selected node (source or starting node) and traverse the graph layer wise thus exploring the neighbour nodes (nodes which are directly

Breadth-First Search (BFS) is an algorithm used for traversing graphs or trees. Traversing means visiting each node of the graph. Breadth-First Search is a recursive algorithm to search all the vertices of a graph or a tree. BFS in python can be implemented by using data structures like a dictionary and lists. Breadth-First Search in tree and graph is almost the same.

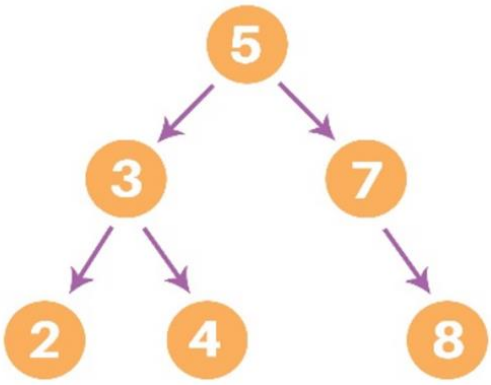
Algorithm:

```
BFS (visited ,G, s)                    //Where G is the graph and s is the source node let Q
        be queue.
        Q.enqueue( s ) //Inserting s in queue until all its neighbor vertices are marked.

        mark s as visited.
        while (Q is not empty)
         //Removing that vertex from queue, whose neighbor will be visited now v =
              Q.dequeue( )

            //processing all the neighbours of v for
            all neighbours w of v in Graph G
                  if w is not visited
                     Q.enqueue( w )//Stores w in Q to further visit its neighbor mark w as
                              visited.
```

Bfs:



```
graph = {
  '5' : ['3','7'],
  '3' : ['2', '4'],
  '7' : ['8'],
  '2' : [],
  '4' : [],
  '8' : []
}
```

```
visited = []
queue = []
```

```
print("Breadth-First Search")
bfs(visited, graph, '5')
```

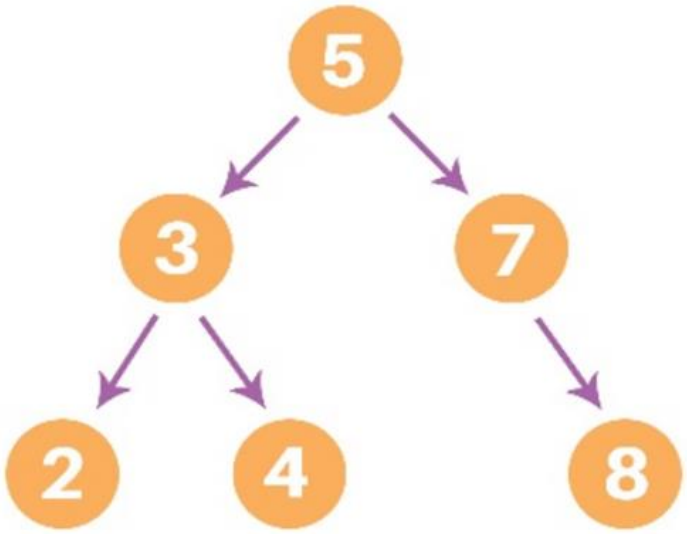
Depth First Search:

As we have already discussed in the class the Depth-First Search (DFS) is an algorithm used for traversing graphs or trees. Traversing means visiting each node of the graph. Depth-First Search is a recursive algorithm to search all the vertices of a graph or a tree. DFS in python can be implemented by using data structures like a dictionary and lists. Dreadth-First Search in tree and graph is almost the same.

Algorithm:

```
DFS(G, u)
   u.visited = true
   for each v ∈ G.Adj[u]
      if v.visited == false
         DFS(G,v)
init() {
   For each u ∈ G
      u.visited = false
    For each u ∈ G
     DFS(G, u)
}
```

Example:



Implementation:

Create a graph/tree in python first, we will create the graph for which we will use the depth-first search. After creation, we will create a set for storing the value of the visited nodes to keep track of the visited nodes of the graph

```
graph = {
  '5' : ['3','7'],
  '3' : ['2', '4'],
  '7' : ['8'],
  '2' : [],
  '4' : [],
  '8' : []
}
```

```
visited = set()
```

After the above process, we will declare a function with the parameters as visited nodes, the graph itself and the node respectively. And inside the function, we will check whether any node of the graph is visited or not using the “if” condition. If not, then we will print the node and add it to the visited set of nodes.
Then we will go to the neighboring node of the graph and again call the DFS function to use the neighbor parameter.

Question:

Ask the user to enter a value and search it in the above tree and print found or not found

Question:

Use the above code to implement the depth limited search limit is 2.