

Practical No: 1

Roll No:

Subject: Artificial Intelligence

Title: Implement DFS and BFS Algorithm. Use an Undirected Graph and develop a Recursive Algorithm for searching all the vertices of the graph or tree data structure.

Program Code :

```
#DFS
graph = {
    '5' : ['3','7'],
    '3' : ['2','4'],
    '7' : ['8'],
    '2' : [],
    '4' : ['8'],
    '8' : []
}
visited = set()
def dfs(visited,graph,node):
    if node not in visited:
        print(node)
        visited.add(node)
        for neighbour in graph[node]:
            dfs(visited,graph,neighbour)

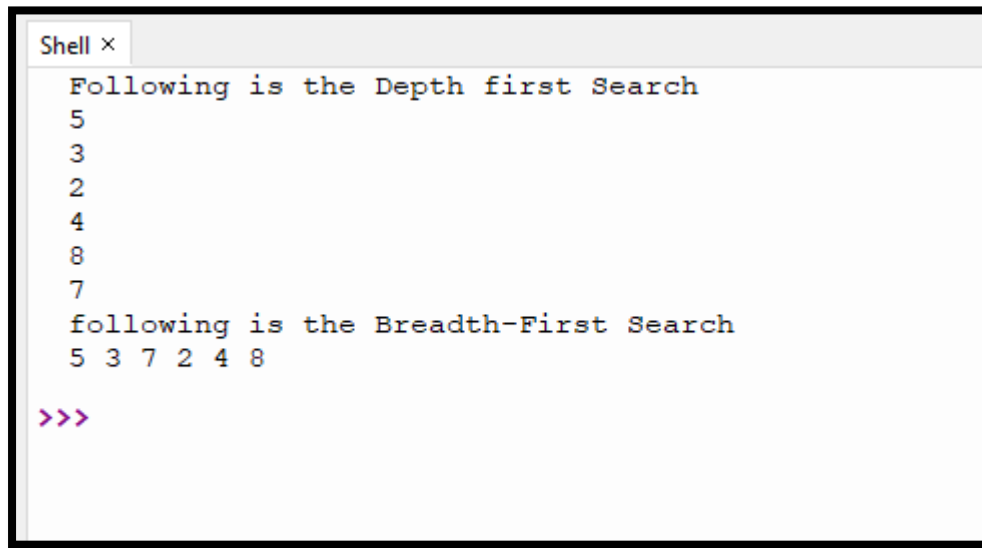
print("Following is the Depth first Search")
dfs(visited,graph,'5')
```

```
# BFS
graph = {
    '5' : ['3','7'],
    '3' : ['2','4'],
    '7' : ['8'],
    '2' : [],
    '4' : ['8'],
    '8' : []
}
visited = []
queue = []
def bfs(visited,graph,node):
    visited.append(node)
    queue.append(node)
    while queue:
        m=queue.pop(0)
        print(m,end=" ")
        for neighbour in graph[m]:
            if neighbour not in visited:
```

```
visited.append(neighbour)
queue.append(neighbour)

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

Output :

A screenshot of a Python Shell window. The window has a title bar that says "Shell x". The output of the code is displayed in a monospaced font. It shows the results of a Depth-First Search (DFS) and a Breadth-First Search (BFS) on a graph. The DFS output lists the nodes 5, 3, 2, 4, 8, and 7 in sequence. The BFS output shows the sequence 5 3 7 2 4 8. The prompt ">>>" is visible at the bottom left of the shell window.

```
Shell x
Following is the Depth first Search
5
3
2
4
8
7
following is the Breadth-First Search
5 3 7 2 4 8
>>>
```