# Artificial Intelligence Lab# 4

**Ques: 1** Consider the following graph:

a) Apply DFS to find to every possible node present in graph. Starting from 1.
b) Find all paths between 1 & 6.
c) Find shortest path between 1 & 6.

Code:

```
graph = {
    '1' : ['2', '3', '4'],
    '2' : ['1', '3', '4'],
    '3' : ['1', '2', '4'],
    '4' : ['1', '2', '3', '5'],
    '5' : ['4', '6', '7', '8'],
    '6' : ['5', '7', '8'],
    '7' : ['5', '6', '8'],
    '8' : ['5', '6', '7'],
}
```

```
def dfs(graph, node, visited):
    if node not in visited:
        visited.append(node)

        for n in graph[node]:
            dfs(graph, n, visited)

    return visited

visited = dfs(graph, '1', [])
print ("Traversal Path : ",visited)
```

```
def dfs_shortest_path(graph, start, goal):
    explored = []
    queue    = [[start]]

    if start == goal:
        return "Thet was easy! Start =  Goal"

    while queue:
        path = queue.pop(0)
        node = path [-1]

        if node not in explored:
            neighbours = graph[node]

            for neighbour in neighbours:
                new_path = list(path)
                new_path.append(neighbour)

                for neighbour in neighbours:
                    new_path = list(path)
                    new_path.append(neighbour)
                    queue.append(new_path)

                    if neighbour == goal:
                        return new_path

            explored.append(node)

    return "So sorry, but a connecting path doesn't exist :("
```

```python
def dfs_paths(graph, start, goal):
    queue = [(start, [start])]

    while queue:
        (vertex, path) = queue.pop(0)

        for next in graph[vertex] - set(path):
            if next == goal:
                yield path + [next]
            else:
                queue.append((next, path + [next]))
```

Answer:

```
F:\8sem\AI\lab_4>python lab4-exercise_78.py
Traversal Path :  ['1', '2', '3', '4', '5', '6', '7', '8']
Shortest path between 1 and 6 :  ['1', '4', '5', '6']
All paths between 1 and 6 :  [['1', '4', '5', '6'], ['1', '3', '4', '5', '6'], ['1', '2', '4', '5', '6']
, ['1', '4', '5', '7', '6'], ['1', '4', '5', '8', '6'], ['1', '3', '2', '4', '5', '6'], ['1', '3', '4',
'5', '7', '6'], ['1', '3', '4', '5', '8', '6'], ['1', '2', '3', '4', '5', '6'], ['1', '2', '4', '5', '7'
, '6'], ['1', '2', '4', '5', '8', '6'], ['1', '4', '5', '7', '8', '6'], ['1', '4', '5', '8', '7', '6'],
['1', '3', '2', '4', '5', '7', '6'], ['1', '3', '2', '4', '5', '8', '6'], ['1', '3', '4', '5', '7', '8',
 '6'], ['1', '3', '4', '5', '8', '7', '6'], ['1', '2', '3', '4', '5', '7', '6'], ['1', '2', '3', '4', '5
', '8', '6'], ['1', '2', '4', '5', '7', '8', '6'], ['1', '2', '4', '5', '8', '7', '6'], ['1', '3', '2',
'4', '5', '7', '8', '6'], ['1', '3', '2', '4', '5', '8', '7', '6'], ['1', '2', '3', '4', '5', '7', '8',
'6'], ['1', '2', '3', '4', '5', '8', '7', '6']]
```

**Ques: 2** Consider the following graph:

a) Apply BFS to find to every possible node present in graph. Starting from A.
b) Find all paths between A & G.
c) Find shortest path between A & G.

```
graph = {
    'A' : ['B', 'C', 'D'],
    'B' : ['A', 'E'],
    'C' : ['A', 'F'],
    'D' : ['A', 'E', 'G'],
    'E' : ['B', 'D', 'G'],
    'F' : ['C', 'G'],
    'G' : ['F', 'E'],
}
```

```
Traversal Path :  ['A', 'B', 'E', 'D', 'G', 'F', 'C']
Shortest path between A and G :  ['A', 'D', 'G']
All paths between A and G : [['A', 'D', 'G'], ['A', 'B', 'E', 'G'], ['A', 'C', 'F', 'G'],
 ['A', 'D', 'E', 'G'], ['A', 'B', 'E', 'D', 'G']]
```