Experiment-1.2

Student Name: Ankit Kumar	**UID: 21BCS5999**
Branch: CSE	**Section/Group: 602-B**
Semester: 5TH	**Date of Performance: 22/8/23**
Subject Name: AIML	**Subject Code: 21CSH-316**

1. **Aim:** Implement the DFS algorithm and analyze its performance and characteristics.

2. **Objective:** To understand the concept of DFS algorithm.

3. **Pseudo Code:**

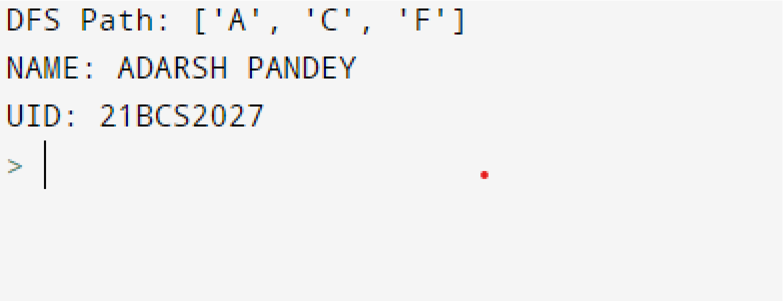
```
function DFS(graph, current, goal, visited):
    if current equals goal:
        return [current]  # Path found, return the single-node path

    if current not in visited:
        add current to visited

        for each neighbor in graph[current]:
            if neighbor not in visited:
                path = DFS(graph, neighbor, goal, visited)
                if path is not null:
                    add current to path
                    return path  # Path found, return the complete path

    return null  # No path found

function DFS_Search(graph, start, goal):
    visited = empty set
    path = DFS(graph, start, goal, visited)
    return path
```

```
# Example graph representation (dictionary adjacency list) graph
= {
    'A': ['B', 'C'],
    'B': ['A', 'D', 'E'],
    'C': ['A', 'F'],
    'D': ['B'],
    'E': ['B', 'F'],
    'F': ['C', 'E']
}

start_node = 'A' goal_node
= 'F'

path = DFS_Search(graph, start_node, goal_node)
print("DFS Path:", path)
```

4. Code:

```
      def dfs(graph, start, goal):
    stack = [(start, [start])]
    visited = set()
    while stack:
        current, path = stack.pop()
        if current == goal:
            return path
        if current not in visited:
            visited.add(current)
            for neighbor in graph[current]:
                if neighbor not in visited:
                    stack.append((neighbor,
             path + [neighbor]))

    return None
graph = {
    'A': ['B', 'C'],
    'B': ['A', 'D', 'E'],
    'C': ['A', 'F'],
    'D': ['B'],
    'E': ['B', 'F'],
    'F': ['C', 'E']
}
start_node = 'A'
goal_node = 'F'
path = dfs(graph, start_node,
        goal_node)
print("DFS Path:", path)
print("NAME: ANKIT KUMAR")
print("UID: 21BCS5999")
```

5. Output:



```
DFS Path: ['A', 'C', 'F']
NAME: ADARSH PANDEY
UID: 21BCS2027
> 
```

6. Learning Outcomes:

The learning outcomes of this experiments are:

- Understood how DFS can be used to find paths between nodes in a graph.
- Learnt how DFS backtracks when exploring paths.
- Understood the concept of DFS.
- Ability to analyze algorithmic performance and optimize graph traversal strategies.