ASSIGNMENT NO. : 1

Name: Tanaya Pradip Bhore
Div.: A   Batch:A
Roll No.:3101016

PROGRAM:

```
from collections import deque

def dfs(visited, graph, node, target):      if node not in
visited:         print(node, end=" ")         visited.add(node)
if node == target:             print(f"\nTarget node '{target}'
found using DFS.")             return True         for neighbour
in graph[node]:
            if dfs(visited, graph, neighbour, target):
                return True
    return False

def bfs(visited, graph, node, target):
    queue = deque([node])
    visited.add(node)

    while queue:
        s = queue.popleft()
print(s,  end="  ")         if
s == target:
            print(f"\nTarget node '{target}' found using BFS.")
            return True        for
neighbour in graph[s]:             if
neighbour not in visited:
visited.add(neighbour)
queue.append(neighbour)      return
False

def main():
    visited1 = set()
visited2 = set()
    graph = {}

    n = int(input("Enter the number of nodes: "))
if n == 0:         print("Graph is empty.")
        return

    for _ in range(n):
        root = input("Enter the root node (character): ")
if root not in graph:
```

```
          graph[root] = []          edges = int(input(f"Enter the number of
child nodes for '{root}': "))        for j in range(edges):            child =
input(f"Enter child {j + 1} for node '{root}': ")
graph[root].append(child)             if child not in graph:
graph[child] = []

   start_node = input("Enter the starting node for traversal: ")
if start_node not in graph:
      print("Invalid starting node.")
      return

   target_node = input("Enter the target node to search for: ")

   print("\nThe following is DFS:")     if not dfs(visited1, graph,
start_node, target_node):         print(f"\nTarget node
'{target_node}' not found using DFS.")

   print("\nThe following is BFS:")     if not bfs(visited2, graph,
start_node, target_node):         print(f"\nTarget node
'{target_node}' not found using BFS.")

if __name___== "__main__":
main()
```

OUTPUT :

```
student@student-OptiPlex-390:~/Downloads$ python3 dfsbfs2.py
Enter the number of nodes: 3
Enter the root node (character): a
Enter the number of child nodes for 'a': 2
Enter child 1 for node 'a': b
Enter child 2 for node 'a': c
Enter the root node (character): d
Enter the number of child nodes for 'd': 2
Enter child 1 for node 'd': e
Enter child 2 for node 'd': f
Enter the root node (character): g
Enter the number of child nodes for 'g': 1
Enter child 1 for node 'g': h
Enter the starting node for traversal: a
Enter the target node to search for: h

The following is DFS:
a b c
Target node 'h' not found using DFS.

The following is BFS:
a b c
```

```
Target node 'h' not found using BFS.
```