Experiment 2

Aim - Implement DFS/DLS/DFID search algorithm in Python.

Code:

```
Tree = {
    "A": ['C', 'B'],
    "B": ['E', 'D'],
    "C": ['G', 'F'],
    "D": ['H'],
    "E": ['J', 'I'],
    "F": ['L', 'K'],
    "G": ['M'],
    "H": [],
    "I": [],
    "J": [],
    "K": [],
    "L": [],
    "M": []
}


def DFS(root, target, stack):
    while len(stack) != 0:
        current_node = stack.pop()
        if current_node == target:
            print("Found goal")
            break
        else:
            children = Tree[current_node]
            stack.extend(children)
        print(stack)


total_depth = 3


def DLS(current, limit, current_depth, goal, stack):
    stack.pop()
    if (current_depth > limit):
        return False
    if current == goal:
        return True
    else:
        children = Tree[current]
        stack.extend(children[::-1])
        print(stack)
        for child in children:
            if DLS(child, limit, (current_depth + 1), goal, stack):
                return True


def IDDFS(goal):
    limit = 0
    found = False
    stack = ['A']
```

```
    while not found:
        print(f"At depth limit {limit}:")
        found = DLS('A', limit, 0, goal, stack)
        stack = ['A']
        limit += 1
        if limit > total_depth:
            print("NOT Exist")
            break
    if found:
        print(f"Found at depth {limit - 1}")

print("Niyati's Code for DFS DLS & IDDFS")

print("The Tree structure is:{Parent:children}")
print(Tree)
want_to_continue = 1
while want_to_continue == 1:
    root_node = input("Enter Root Node: ")
    goal_node = input("Enter Goal Node: ")
    user_inp = input("What algorithm to use? Press 1 for DFS, 2 for DLS and 3 for IDDFS: ")
    stack = ['A']
    print(stack)
    if user_inp == '1':
        DFS(root_node, goal_node, stack)
        stack = ['A']
    elif user_inp == '2':
        limit = int(input("Enter depth limit: "))
        if DLS(root_node, limit, 0, goal_node, stack):
            print("Found within given depth")
        else:
            print("Not Found within given depth")
        stack = ['A']
    elif user_inp == '3':
        IDDFS(goal_node)
    else:
        print("Enter a valid number")
        stack = ['A']

    want_to_continue = int(input("Press 1 to continue and anything else to exit: "))
```

Output:

Niyati's Code for DFS DLS & IDDFS

The Tree structure is:{Parent : children}

{'A': ['C', 'B'], 'B': ['E', 'D'], 'C': ['G', 'F'], 'D': ['H'], 'E': ['J', 'I'], 'F': ['L', 'K'], 'G': ['M'], 'H': [], 'I': [], 'J': [], 'K': [], 'L': [], 'M': []}

Enter Root Node: A

Enter Goal Node: G

What algorithm to use? Press 1 for DFS, 2 for DLS and 3 for IDDFS: 1

['A']

['C', 'B']

['C', 'E', 'D']

['C', 'E', 'H']

['C', 'E']

['C', 'J', 'I']

['C', 'J']

['C']

['G', 'F']

['G', 'L', 'K']

['G', 'L']

['G']

Found goal

Press 1 to continue and anything else to exit: 1

Enter Root Node: A

Enter Goal Node: G

What algorithm to use? Press 1 for DFS, 2 for DLS and 3 for IDDFS: 2

Enter depth limit: 1

['A']

['B', 'C']

['B', 'F', 'G']

['D', 'E']

Not Found within given depth

Press 1 to continue and anything else to exit: 1

Enter Root Node: A

Enter Goal Node: G

What algorithm to use? Press 1 for DFS, 2 for DLS and 3 for IDDFS: 3

At depth limit 0:

['A']

['B', 'C']

At depth limit 1:

['A']

['B', 'C']

['B', 'F', 'G']

['D', 'E']

At depth limit 2:

['A']

['B', 'C']

['B', 'F', 'G']

Found at depth 2

Press 1 to continue and anything else to exit: 0