Lab 3

15/07/2024

Brief Summary:

The script addresses the following tasks and problems:

- **DFS Recursive Function (`dfs_recursive`)**:

  - Uses recursion to explore the graph depth-first starting from a given node (`start`).
  - Maintains a `visited` set to track visited nodes and avoid revisiting.
  - Prints each node in the order they are visited, recursively visiting each neighbor not yet visited.

- **BFS Function (`bfs`)**:

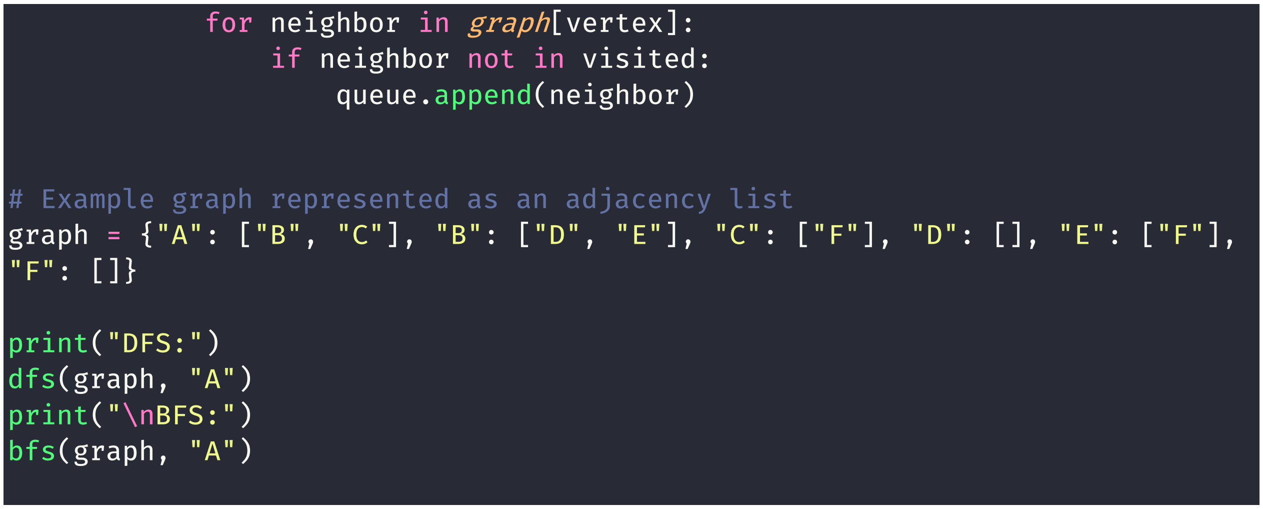
  - Implements BFS using a queue (`deque`) starting from a given node (`start`).
  - Uses a `visited` set to track visited nodes and ensure each node is processed once.
  - Prints each node in the order they are visited, expanding from the current node to its neighbors level by level.

Code:

```
from collections import deque


def dfs(graph, start, visited=None):
    if visited is None:
        visited = set()
    visited.add(start)
    print(start, end=" ")
    for neighbor in graph[start]:
        if neighbor not in visited:
            dfs(graph, neighbor, visited)


def bfs(graph, start):
    visited = set()
    queue = deque([start])
    while queue:
        vertex = queue.popleft()
        if vertex not in visited:
            visited.add(vertex)
            print(vertex, end=" ")
```

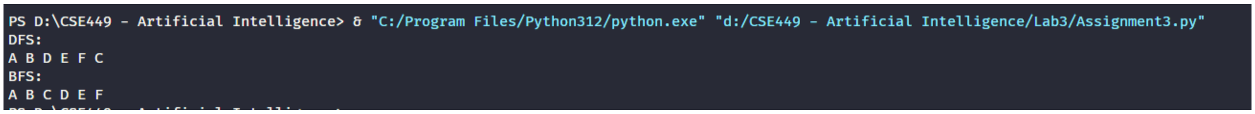


```
            for neighbor in graph[vertex]:
                if neighbor not in visited:
                    queue.append(neighbor)


# Example graph represented as an adjacency list
graph = {"A": ["B", "C"], "B": ["D", "E"], "C": ["F"], "D": [], "E": ["F"],
"F": []}

print("DFS:")
dfs(graph, "A")
print("\nBFS:")
bfs(graph, "A")
```

Output:



```
PS D:\CSE449 - Artificial Intelligence> & "C:/Program Files/Python312/python.exe" "d:/CSE449 - Artificial Intelligence/Lab3/Assignment3.py"
DFS:
A B D E F C
BFS:
A B C D E F
PS D:\CSE449 - Artificial Intelligence>
```