SUPERIOR UNIVERSITY

Artificial Intelligence (Lab)

Assignment - 6

Name:

Ali Maqsood.

Roll no:

SU92-BSAIM-F23-050.

Department:

Software Engineering Department.

Program:

Artificial Intelligence.

Section:

BSAI-3A

Question # 1:

BFS without Queue & without Node.

Explanation:

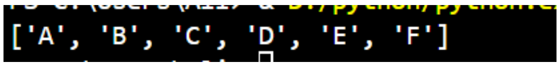
This code finds all the connected nodes in a graph using the **Breadth-First Search (BFS) algorithm**. It starts from a given node (A) and explores all its direct connections before moving deeper. It keeps track of **visited nodes** to avoid repetition and processes nodes level by level using current_level and next_level. The search continues until all reachable nodes are visited, and then it prints the order in which the nodes were explored.

Code:

```
def task1():
    graph={
        "A":["B","C"],
        "B":["D","E"],
        "C":["F"],
        "D":[],
        "E":["F"],
        "F":[]
    }
    def bfs(graph, start):
        visited_nodes = []
        current_level = [start]

        while current_level:
            next_level = []
            for node in current_level:
                if node not in visited_nodes:
                    visited_nodes.append(node)
                    next_level.extend(graph[node])
            current_level = next_level
        print (visited_nodes)
    bfs(graph, "A")
# task1()
```

Output:



```
['A', 'B', 'C', 'D', 'E', 'F']
```

Question # 2:

BFS with Queue & Node.

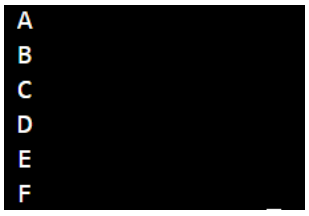
Explanation:

This is a simple implementation of (Breath First Search) on a graph A → F. It uses a queue to add all the nodes one by one and picks one node from the start of the queue, then print that, add it in visited and add its related nodes in the queue to visited for later. This process goes on and on with while loop until the each node is visited to the end of the tree.

Code:

```
def task2():
    graph={
        "A":["B","C"],
        "B":["D","E"],
        "C":["F"],
        "D":[],
        "E":["F"],
        "F":[]
    }
    def bfs(graph,start):
        visited=[]
        queue=[start]
        while queue:
            current=queue.pop(0)
            print(current,end='')
            if current in graph:
                for i in graph[current]:
                    if i not in visited:
                        queue.append(i)
                    visited.append(i)
            print()
    bfs(graph,"A")
# task2()
```

Output:



```
A
B
C
D
E
F
```