Esha Asif
034
BSAI-3A
Task 6
AI LAB

Task1:BFS without queue and node

```
#def bfs(graph, start, visited=None):
#     if visited is None:
#         visited = set()

#     print(start,end=" ")
#     visited.add(start)

#     for neighbor in graph[start]:
#         if neighbor not in visited:
#             bfs(graph, neighbor, visited)
# graph = {
#     0: [1, 2],
#     1: [0, 3, 4],
#     2: [0, 5],
#     3: [1],
#     4: [1],
#     5: [2]}

# start_node = 0
# print("bfs without queue and node")
# bfs(graph, start_node)
```

```
bfs without queue and node
0 1 3 4 2 5
```

Task2:Bfs with node and queue

```
#class Node:
#     def __init__(self, value):
#         self.value = value
#         self.neighbors = []
```

```
# def bfs_with_queue_and_node(start_node):
#     visited = set()
#     queue = deque([start_node])
#     visited.add(start_node)

#     while queue:
#         current_node = queue.popleft()
#         print(current_node.value, end=" ")

#         for neighbor in current_node.neighbors:
#             if neighbor not in visited:
#                 visited.add(neighbor)
#                 queue.append(neighbor)


# node0 = Node(0)
# node1 = Node(1)
# node2 = Node(2)
# node3 = Node(3)
# node4 = Node(4)
# node5 = Node(5)

# node0.neighbors = [node1, node2]
# node1.neighbors = [node0, node3, node4]
# node2.neighbors = [node0, node5]
# node3.neighbors = [node1]
# node4.neighbors = [node1]
# node5.neighbors = [node2]

# print("bfs with queue and node")
# bfs_with_queue_and_node(node0)
```

```
bfs with queue and node
0 1 2 3 4 5
```