# Breadth-First Search (BFS) Implementations

## Recursive BFS Implementation

# Recursive BFS Implementation  
def BFS(tree, root, goal):  
 result = []  
  
 def traverse(nodes):  
 if not nodes:  
 return  
 next\_nodes = []  
 for node in nodes:  
 result.append(node)  
 if node == goal:  
 return  
 for child in tree.get(node, []):  
 next\_nodes.append(child)  
 traverse(next\_nodes)  
  
 traverse([root])  
 return result

## Iterative BFS Implementation

# Iterative BFS Implementation  
def bfs(graph, start, goal):  
 queue = [start]  
 visited = []  
  
 while queue:  
 node = queue.pop(0)  
 visited.append(node)  
  
 if node == goal:  
 print("Found the goal:", node)  
 return visited  
  
 for neighbor in graph.get(node, []):  
 if neighbor not in visited and neighbor not in queue:  
 queue.append(neighbor)  
 return visited

## Example Graph and Test

# Example graph  
graph = {  
 'A': ['B', 'C'],  
 'B': ['D', 'E'],  
 'C': ['F'],  
 'D': [],  
 'E': ['G'],  
 'F': [],  
 'G': []  
}  
  
# Test recursive BFS  
print("Recursive BFS Result:", BFS(graph, 'A', 'F'))  
  
# Test iterative BFS  
print("Iterative BFS Result:", bfs(graph, 'A', 'F'))