PRACTICAL 4

Topic: BFS

Introduction: Breadth-First Search (BFS)

Breadth-First Search (BFS) is a graph traversal algorithm that explores all nodes at the present depth level before moving on to nodes at the next depth level. It works systematically by visiting neighbors level by level, making it ideal for finding the shortest path in an unweighted graph.

Key Characteristics

1. Exploration Level by Level:
   * Nodes are explored layer by layer starting from a source node.
2. Queue Data Structure:
   * A queue is used to keep track of nodes to be explored.
3. Shortest Path Guarantee:
   * BFS guarantees the shortest path in terms of the number of edges for unweighted graphs.

Code:

from collections import deque

def bfs(graph, start):

    """Perform BFS traversal from the starting node."""

    visited = set()           # Set to keep track of visited nodes

    queue = deque([start])    # Queue for BFS

    result = []               # List to store traversal order

    while queue:

        node = queue.popleft()

        if node not in visited:

            visited.add(node)         # Mark node as visited

            result.append(node)       # Add to result

            queue.extend(graph[node]) # Add unvisited neighbors to the queue

    return result

# Example graph as an adjacency list

graph = {

    'A': ['B', 'C'],

    'B': ['A', 'D', 'E'],

    'C': ['A', 'F'],

    'D': ['B'],

    'E': ['B', 'F'],

    'F': ['C', 'E']

}

# Perform BFS starting from 'A'

bfs\_result = bfs(graph, 'A')

print("BFS Traversal:", bfs\_result)

Output:

