from collections import deque

# BFS Function

def bfs(graph, start):

    visited = set()

    queue = deque([start])

    order = []

    while queue:

        node = queue.popleft()

        if node not in visited:

            visited.add(node)

            order.append(node)

            for neighbor in graph[node]:

                if neighbor not in visited:

                    queue.append(neighbor)

    return order

# DFS Function

def dfs(graph, start):

    visited = set()

    stack = [start]

    order = []

    while stack:

        node = stack.pop()

        if node not in visited:

            visited.add(node)

            order.append(node)

            for neighbor in reversed(graph[node]):  # reverse for correct DFS order

                if neighbor not in visited:

                    stack.append(neighbor)

    return order

# ---- Main Program ----

if \_\_name\_\_ == "\_\_main\_\_":

    # Example Graph (Adjacency List)

    graph = {

        'A': ['B', 'C'],

        'B': ['A', 'D'],

        'C': ['A', 'D'],

        'D': ['B', 'C']

    }

    start = 'A'

    bfs\_order = bfs(graph, start)

    dfs\_order = dfs(graph, start)

    print("Graph:", graph)

    print("Start Node:", start)

    print("BFS Order:", bfs\_order)

    print("DFS Order:", dfs\_order)