Problem – 6 : Topological Sort (BFS)

```
from collections import defaultdict, deque

def topological_sort_bfs(graph):
    in_degree = {node: 0 for node in graph}
    for node in graph:
        for neighbor in graph[node]:
            in_degree[neighbor] += 1

    queue = deque()
    for node in in_degree:
        if in_degree[node] == 0:
            queue.append(node)

    topological_order = []
    while queue:
        node = queue.popleft()
        topological_order.append(node)
        for neighbor in graph[node]:
            in_degree[neighbor] -= 1
            if in_degree[neighbor] == 0:
                queue.append(neighbor)

    if len(topological_order) != len(graph):
        # The graph contains cycles, so topological sort is not possible
        return None

    return topological_order

graph = {
    'A': ['B', 'C'],
```

```
    'B': ['D'],
    'C': ['D'],
    'D': ['E'],
    'E': []
}

result = topological_sort_bfs(graph)
if result is not None:
    print("Topological Sort (BFS):", result)
else:
    print("The graph contains cycles. Topological Sort is not possible.")
```