Problem – 7 : Topological Sort (DFS)

```
from collections import defaultdict


class Graph:
    def __init__(self):
        self.graph = defaultdict(list)


    def add_edge(self, u, v):
        self.graph[u].append(v)


    def topological_sort_util(self, v, visited, stack):
        visited[v] = True


        for neighbor in self.graph[v]:
            if not visited[neighbor]:
                self.topological_sort_util(neighbor, visited, stack)


        stack.append(v)


    def topological_sort(self):
        visited = [False] * (max(self.graph) + 1)
        stack = []


        for i in range(len(visited)):
            if not visited[i]:
                self.topological_sort_util(i, visited, stack)


        return stack[::-1]


g = Graph()
g.add_edge(5, 2)
```

```
g.add_edge(5, 0)
g.add_edge(4, 0)
g.add_edge(4, 1)
g.add_edge(2, 3)
g.add_edge(3, 1)

print("Topological Sort (DFS):", g.topological_sort())
```