Problem 1 : Strongly Connected Components – Kosaraju's Algorithm: G-54 Problem Statement: Given a Directed Graph with V vertices (Numbered from 0 to V-1) and E edges, Find the number of strongly connected components in the graph.

```
from collections import defaultdict


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


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


    def dfs(self, v, visited, stack):
        visited[v] = True
        for i in self.graph[v]:
            if not visited[i]:
                self.dfs(i, visited, stack)
        stack.append(v)


    def transpose(self):
        g = Graph(self.V)
        for i in self.graph:
            for j in self.graph[i]:
                g.add_edge(j, i)
        return g


    def count_scc(self):
        stack = []
        visited = [False] * self.V
        for i in range(self.V):
```

```
            if not visited[i]:
                self.dfs(i, visited, stack)

        transposed_graph = self.transpose()

        visited = [False] * self.V
        scc_count = 0

        while stack:
            v = stack.pop()
            if not visited[v]:
                transposed_graph.dfs(v, visited, [])
                scc_count += 1

        return scc_count


V = 5
graph = Graph(V)
graph.add_edge(0, 1)
graph.add_edge(1, 2)
graph.add_edge(2, 0)
graph.add_edge(1, 3)
graph.add_edge(3, 4)
num_scc = graph.count_scc()
print("Number of Strongly Connected Components:", num_scc)
```