1. **Programming Question** (15 marks) Implement Kosaraju’s strongly connected component algorithm in C++ or Python. Specifically, your program should take a directed graph as input and outputs the sizes (in terms of number of nodes) of the five largest strongly connected components, listed in order from largest to smallest. If there are less than 5 components in the graph, you should output 0 for the remaining components. The following is the format of the input file your program will get:
   
   line 1: n (number of nodes, an integer)
   line 2: m (number of edges, an integer)
   line 3 to m+2: each line contains a directed edge described as two integers u and v.

   The nodes are numbered from 1 to n. u is the origin of the edge and v its destination.

   Your program should write as output five integers, separated by a tab, corresponding to the number of nodes in each of the five largest strongly connected components. You can assume there are no self loops (u, u) and multiple parallel edges ((u, v) and (u, v)) in the input file.

   As an example consider the following sample input:

   6
   9
   1 2
   1 3
   2 3
   2 5
   3 1
   3 4
   4 5
   4 6
   5 4

   Your output should be:

   3 2 1 0 0

   For submissions instructions, please follow the same information that was posted on Piazza for the previous programming question from assignment 3.
2. **Scorpion Graph** (8 marks) Scorpion graph is an undirected graph with $n$ nodes, where one node (the body) had degree $n-2$, connecting to $n-3$ nodes (feet) and to the tail. The tail has degree 2, and connects to sting, which has degree 1. The feet may or may not be connected to other feet, but not to tail or sting. See Figure 1

Given a graph $G$ represented by an adjacency matrix, give an algorithm to decide if $G$ is a scorpion graph in $O(n)$ time.

![Figure 1: A scorpion graph](image)

3. **Topological Sorting** (9 marks) Let $G = (V,E)$ be a weighted, directed acyclic graph (DAG) with $n$ vertices and $m$ edges, where $m \geq n$. Let $s,t \in V$. We say that a path $v_1, v_2, ..., v_l$ is monotone if $w(v_1, v_2) \leq w(v_2, v_3) \leq \cdots \leq w(v_{l-1}, v_l)$. We want to compute a shortest monotone path from $s$ to $t$, i.e., a monotone path of the smallest total weight. Give an $O(mn)$-time algorithm using dynamic programming. Your algorithm should return the optimal path in addition to the optimal value. If no such path exists, you can just return “no”.

[Hint: Consider topologically sorting $G$.]

4. **MST** (9 marks) Let $G = (V,E)$ be a weighted undirected connected graph, where all the edge weights are distinct. Let $T^*$ denote the minimum spanning tree.

   (a) (3 pt) Prove that if $C$ is a cycle in $G$ and $e$ is the largest-weight edge in $C$, then $e$ cannot be in $T^*$.

   (b) (3 pt) Suppose that $G$ has $m \leq n + 157$ edges. For this special case, give an MST algorithm that runs in $O(n)$ time beating Kruskals and Prims algorithm (which would run in $O(n \log(n))$ time).

   [Hint: use part (a).]

   (c) (3 pt) Consider the following problem: Given $s, t \in V$, find a path from $s$ to $t$ that minimizes the largest edge weight along the path. Note that there could be more than
one optimal solution to this problem. Prove that we can find one optimal solution to the problem by simply computing the MST $T^*$ and returning the path from $s$ to $t$ in the tree $T^*$. [Hint: use part (a).]

5. [**k-coloring** (9 marks)] An undirected graph $G(V, E)$ is $k$-colorable if and only if each vertex $v \in V$ can be labeled with one of $k$ colors such that for each edge $(u, v) \in E$ $u$ and $v$ get different colors. Let $G$ be an undirected graph with vertices $v_1, \ldots, v_n$. We say $G$ has width $w$ if we can order the vertices in a sequence $v_{i_1}, v_{i_2}, v_{i_3}, \ldots, v_{i_n}$ such that each $v_{i_j}$ has $\leq w$ edges to vertices $v_{i_1}, \ldots, v_{i_{j-1}}$. Prove by induction that every graph with width $\leq w$ is $(w + 1)$-colorable.