1. (15 points) Run the strongly connected components algorithm on the following directed graph $G$. When doing DFS on $G^R$, whenever there is a choice of vertices to explore, always pick the one that is alphabetically first.

\[
\begin{align*}
A &: B, D \\
B &: C, D, E \\
C &: F \\
D &: E \\
E &: A, C \\
F &: I, J \\
G &: A, D, K \\
H &: D, E, G, I \\
I &: J \\
J &: C \\
K &: H
\end{align*}
\]

Answer the following questions:

1. In what order are the strongly connected components (SCCs) found?
2. Which are source SCCs and which are sink SCCs?
3. Draw the ”metagraph” (each meta-node is an SCC of $G$)

2. (15 points) Consider the following problem: Given a directed graph $G = (V, E)$, if there exists a vertex in $V$ that can reach all other vertices in $V$ then return TRUE. Otherwise return FALSE.

Consider the following algorithm that claims to solve this problem:

**Input:** $G = (V, E)$

- Run DFS on the graph $G$ and keep track of post numbers.
- Run explore (graphsearch) on $G$ starting at the vertex $h$ with the highest post number.
- If all vertices are visited return TRUE
- Otherwise return FALSE.

(a) State whether this algorithm is correct or not.
(b) Justify part (a). (If you say that it works, prove the correctness of the algorithm. If you say that it does not work then provide a counterexample.)

3. (20 points) Suppose you had $n$ matrices with dimensions: $a_1 \times b_1, a_2 \times b_2, \ldots, a_n \times b_n$. Your goal is to determine, given two integers $s$ and $t$, whether it is possible to multiply a sequence from the list of given matrices together, in any order and possibly not using all of the matrices, to end up with a matrix with dimensions $s \times t$.

For example, if the list of matrix dimensions is $A : 3 \times 5, B : 5 \times 7, C : 7 \times 9, D : 9 \times 5, E : 9 \times 3$, and $F : 7 \times 5$ we can construct a $9 \times 9$ matrix as $D \ast B \ast C$. Discuss runtime and correctness.
4. (25 points) You are given a directed graph in which each node \( u \in V \) has an associated price \( p_u \) which is a positive integer. Define the array cost as follows for each \( u \in V \),

\[
\text{cost}[u] = \text{Price of the cheapest node reachable from } u \text{ (including } u \text{ itself)}. \]

Your goal is to compute the cost for each vertex in \( V \).

(a) Give a linear time algorithm that works for DAG’s (Hint: Handle the vertices in a particular order)

(b) Extend this to a linear time algorithm that works for any directed graph (Hint: Recall the two-tiered structure of directed graphs)

Give a run-time analysis and proof of correctness for both parts.

5. (25 points) Give an efficient algorithm that takes as input a directed graph \( G = (V, E) \) with edges colored either blue or red and a starting vertex \( s \) and outputs the list of all vertices \( t \) where there is a path (not necessarily simple) that goes from \( s \) to \( t \) and alternates between blue and red edges. Discuss run-time and proof of correctness.