PROBLEM 1. *Short answer*

(a) *State the max-flow min-cut theorem.*

(b) *State the cut lemma.*

(c) *The method by which augmenting paths are chosen in the Ford-Fulkerson algorithm does not impact its running time.*

(d) *Alice tells Bob that Vertex Cover doesn't have a polynomial-time solution because it is NP-complete. Bob says he has doubts. Who is right?*

(e) *Every problem in NP is also NP-complete.*

(f) *In every instance of the Stable Matching Problem, there is a stable matching containing a pair (m, w) such that m is ranked first on the preference list of w and w is ranked first on the preference list of m. Explain why, or provide a counter-example.*

(g) *Let $G = (V, E, c)$ be a network flow graph with source s and sink t, and let P be any directed path in G from s to t. Suppose we increase the capacity of every edge in P by one unit. Can the max flow increase by more than 1 ?*

PROBLEM 2. *Equity*

Let $G = (V, E)$ be a connected graph with distinct positive edge costs, and let T be a spanning tree of G, but not necessarily a *minimum spanning tree*. The *heavy edge* of a spanning tree T is the edge with the largest cost. A spanning tree is said to be *equitable* if there is no other spanning tree with a lighter heavy edge. In other words, such a tree minimizes the cost of the most costly edge (instead of minimizing the overall cost).

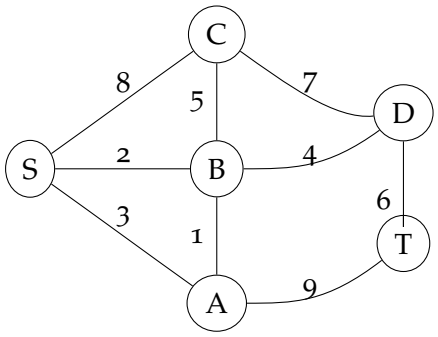
(a) Is every minimum spanning tree of G an equitable spanning tree of G?

(b) Is every equitable spanning tree of G a minimum spanning tree?

PROBLEM 3. *Another all pairs shortest path*

Set $\text{BSHORT}_{i,j,k}$ to be the shortest path from i to j that uses at most k edges. Note this is different from the ASHORT variable that we used in class in that we do not restrict the intermediate nodes to be $1 \ldots k$ in this formulation. State a recursive formula for BSHORT. Devise an algorithm that uses this recurrence. What is the running time of this algorithm?

PROBLEM 4. *Basic Algorithm*



(a) Execute Dijkstra's algorithm to find the shortest path from S to every other vertex on the following graph. Give the final output graph with the shortest distance to each node from s and the order in which edges are added to the output graph.

(b) Execute Kruskal's algorithm to find the minimum spanning tree on the following graph. Give the final output tree and the order in which edges are added to the output tree.

(c) Execute Prim's algorithm.

PROBLEM 5. *Stable Matching*

P1: $1 > 2 > 3$	R1: $2 > 3 > 1$
P2: $2 > 3 > 1$	R2: $3 > 1 > 2$
P3: $3 > 1 > 2$	R3: $1 > 2 > 3$

Find a stable matchings for the preferences listed above.

PROBLEM 6. *Reductions*

You are given a directed graph $G = (V, E)$, where each edge has a positive integer weight $w(u, v)$. You want to find the shortest path between a given pair of vertices s and t, but if there are multiple shortest paths, you want to return the one with the fewest number of edges. You have been given code for a highly optimized version of Dijkstra written in a programming language that abhi invented, and he is the only one who understands its arcane rules, so you are unlikely to succeed at changing the implementation. How can you solve the problem?