Practice for Midterm 2.

The following instructions will also be on the exam:

During this exam you may not use electronic devices, books, lecture notes, or any other source that may assist in solving the questions.

Unless stated otherwise, you do not need to repeat any proofs or running time analysis that we did in class. For example, you may sort in $O(n \log n)$ time without describing how. You may not use algorithms that we did not learn, such as hash tables or DFS.

Unless stated otherwise, all graphs are simple and undirected. Running times should use $O(\cdot)$-notation.

Each problem is worth the same number of points. Only your top 5 answers will be used, so start with the ones that seem easier!

The problems below are for practicing the material for midterm 2. The level is a bit harder than the ones on the midterm. Questions marked “difficult” are too difficult for the midterm. The midterm consists of exactly 6 problems.

**Problem 1.** Draw a bipartite graph with vertex priorities. The graph should have (at least) two stable perfect matchings.

**Problem 2.** Let $G = (V, E)$ be a graph with positive edge weights. We define the *multiplicative* weight of a spanning tree to be the product of its edge weights. Describe an algorithm for finding a minimum spanning tree, with respect to multiplicative weights. The running time should be $O(|V| \log |V| + |E|)$.

Hint: This requires a trick we learned in the “lightest paths” topic.

**Problem 3.** Explain why every tree contains at most one perfect matching.

**Problem 4.** (This problem will not be on exam, since understanding what it is asking for is too confusing.) We wish to rebuild the communication network of a company. We have a list of computers and routers, and the cost of connecting every two. We wish to find the cheapest connected network possible. As before, a router can be connected to any number of computers and routers. On the other hand, a computer can only be connected to a single router (two computers cannot be directly connected).

(a) Rewrite the problem as a graph problem.
(b) Describe an algorithm for solving the problem in time $O(|V| \log |V| + |E|)$.
**Problem 5.** Let \( G = (V, E) \) be a directed graph with edge weights and let \( s \) be a vertex of \( V \) (weights might be negative). We are told that the lightest path in this graph is of **length** at most \( \log(|V|) \). Be careful when reading the previous sentence: The lightest paths are defined with respect to the weights, but the length of a path is the number of edges in it.

Describe an algorithm that finds lightest paths from \( s \) with running time \( O(|E| \log |V| + |V|) \).

Hint: Use Bellman-Ford.

**Problem 6.** Let \( G = (V, E) \) be a graph with edge weights and let \( s \) be a vertex of \( V \) of degree at most 20. Describe an algorithm that finds the lightest among the spanning trees of \( G \) in which \( s \) has degree at most 2. The running time should be \( O(|V| \log |V| + |E|) \).

For example, the graph in the figure has an MST of weight 3, but the lightest spanning tree with the degree restriction is of weight 4.

**Problem 7.**

(a) Let \( G = (V, E) \) be a graph where all the vertex degrees are 2. Describe what \( G \) consists of. (In the matching class we considered graphs with degrees 0,1,2).

(b) Let \( H \) be a bipartite graph and let \( M \) and \( M' \) be perfect matchings in \( H \). No edge repeats in both matchings. Let \( G^* = (V, E^*) \) be the graph consisting only of the edges of \( M \) and \( M' \). Describe \( G^* \). Your answer should be “\( G^* \) consists of ...” (one sentence is enough).

**Problem 8.** Does there exist a perfect stable matching where each hospital has their least favorite student? If so, provide an example of such a matching. If not, explain why not.

**Problem 9.** Let \( G = (V, E) \) be a directed graph with edge weights. When ignoring the edge directions, \( G \) does not contain any cycles. Describe an algorithm that receives vertices \( s, t \) and finds the lightest path from \( s \) to \( t \). The running time should be better than that of Dijkstra’s algorithm.

**Problem 10.** Adam is playing a game with TiM. They receive a bipartite \( G = (V_1, V_2, E) \) with \( |V_1| = |V_2| \). Tim starts by choosing a vertex \( u_1 \) from \( V_1 \). Adam then travels from \( u_1 \) across an edge to a vertex \( u_2 \) of \( V_2 \). Then Tim travels from \( u_2 \) across an edge to a vertex \( u_3 \). The players continue to travel the graph, and they are not allowed to revisit a vertex that already appeared in the path. The player that gets stuck without a valid edge to cross loses (and must do student advising for a week!)

We know that \( G \) does not contain a perfect matching. Show that there is a strategy that Tim can use to always win. No matter what Adam does.
Hint. Tim should start by finding a maximum matching $M$, and choose $u_1$ to be a vertex that is not matched in $M$. What property of a maximum matching might be helpful here?

**Problem 11.** Let $G = (V, E)$ be a graph with edge weights. Describe an algorithm that receives an edge $e$ of $E$ and checks whether there exists an MST of $G$ that contains $e$. The running time should be $O(|V| \log |V| + |E|)$.

**Problem 12.** Difficult Problem!
Solve problem 11 and problem 4 from HW4 with running time $O(|V| + |E|)$.

**Problem 13.** Let $G = (V, E)$ be a graph and let $w_1, w_2 : E \to \mathbb{R}$ be two weight function over the edges. That is, every edge $e$ has two weights – one is $w_1(e)$ and the other is $w_2(e)$. All weights are positive. Write an algorithm that receives vertices $s$ and $t$. Out of all the lightest paths from $s$ to $t$ according to $w_1$, the algorithm should return the lightest path according to $w_2$. The running time should be $O(|V| \log |V| + |E|)$.

In the figure, each edge has two weights – the first is $w_1(e)$ and the second is $w_2(e)$. There are three lightest paths from $s$ to $t$ according to $w_1(\cdot)$, each of weight 8. Out of those, one has $w_2(\cdot)$ weight of 9, another has $w_2(\cdot)$ weight of 6, and the last one 8. Thus, the algorithm should find the path with $w_1(\cdot)$ weight of 8 and has $w_2(\cdot)$ weight of 6.

**Problem 14.** Let $G = (V, E)$ be a directed graph with non-negative edge weights. Every edge is either red or blue. Describe an algorithm that receives vertices $s, t$ and finds a lightest path from $s$ to $t$ out of the paths that pass through a red edge a number of times that is a multiple of three. The running time should be $O(|V| \log |V| + |E|)$.

In the figure, the lightest path from $s$ to $t$ that satisfies the above is of weight 24.

**Problem 15.** Let $G = (V, E)$ be a graph with edge weights, such that no two edges have the same weight. We say that an edge $e$ is heavy if it is the heaviest edge on some cycle of $G$. Describe an algorithm that finds all the heavy edges in $G$ in time $O(|V| \log |V| + |E|)$.

**Problem 16.** Is the following true or false? If true, explain why. If false, give a counterexample.
Let $G = (V, E)$ be a graph with edge weights. Let $T$ be an MST of $G$. Then after increasing the weight of each edge by 1, the tree $T$ always remains an MST.
For each of the following statements, state whether it is true or false. Explain your answers.

- Both BFS and Dijkstra are algorithms for finding shortest paths, so they both answer the same problems.
- Every graph with $|V| - 10$ edges is not connected.
- Every graph with $100|V|$ edges is connected.
- Every lightest paths tree is an MST.
- Let $G = (V, E)$ be a graph with distinct edge weights (no weight occurs twice). Then the MST consists of the $|V| - 1$ lightest edges of $E$.
- Given a graph $G = (V, E)$ with edge weights and a vertex $s$, there is always an MST of $G$ where $s$ is a leaf.