Final Exam

- Do not open this quiz booklet until you are directed to do so. Read all the instructions first.
- The quiz contains 8 problems. You have 180 minutes to earn 180 points.
- This quiz booklet contains 21 pages (including this one) plus one sheet of scratch paper.
- This quiz is closed book. You may use two double-sided letter (8.5'' × 11'') or A4 crib sheet. No calculators or programmable devices are permitted. Cell phones must be put away.
- Write your solutions in the space provided. If you run out of space, continue your answer on the back of the same sheet and make a notation on the front of the sheet.
- Write your name on every page of the exam (in case it falls apart).
- If you leave the answer to a problem part blank, you will receive a score of approximately 20%. If you write a bad answer, your score may reduce to 0.
- Many problems begin with a paragraph in italics. You may skip these paragraphs.
- Do not waste time deriving facts that we have studied. Just cite results from class.
- Do not spend too much time on any one problem. Generally, a problem’s point value is an indication of how many minutes to spend on it.
- Show your work, as partial credit will be given. You will be graded not only on the correctness of your answer, but also on the clarity with which you express it. Please be neat.
- Good luck!

<table>
<thead>
<tr>
<th>Problem</th>
<th>Title</th>
<th>Points</th>
<th>Parts</th>
<th>Grade</th>
<th>Initials</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Name</td>
<td>5</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>True or False</td>
<td>32</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Short Answer</td>
<td>48</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Final Draft</td>
<td>15</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Final Countdown</td>
<td>15</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Final Act</td>
<td>15</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Final Project</td>
<td>15</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Final Destination(s)</td>
<td>20</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Final Boss</td>
<td>15</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>180</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Name: ____________________________
Problem 0. [5 points] Write your name on every page of this exam. (The 20% rule does not apply to this question!)

This exam is calibrated so that you should spend about one minute per point. This question is worth 5 points because we recommend that you spend about 5 minutes skimming all the questions as you write your name.

Problem 1. True or False [32 points] (8 parts)

Circle T or F for each of the following statements to indicate whether the statement is true or false and briefly explain why. A wrong answer is worth 0 points; a blank answer is worth 1 point; a correct answer with a blank explanation is worth 1 point; a correct answer with a nonsensical explanation is worth 0 points; a correct answer with a good explanation is worth 4 points.

(a) T F If every edge in a directed acyclic graph has an arbitrary weight, then one can find the simple path from node $s$ to node $t$ of maximum total weight by negating the weight on each edge and running a shortest path algorithm.

(b) T F If every edge in a directed graph has a nonnegative weight, then one can find the simple path from node $s$ to node $t$ of maximum total weight by negating the weight on each edge and running a shortest path algorithm.
(c) T F Assume random variables $X_1, X_2, \ldots, X_n$ are independent and uniform over $[-1, 1]$. By Markov’s inequality, for any $k > 0$,

$$\Pr \left\{ \sum_{i=1}^{n} X_i \geq k \cdot \mathbb{E} \left[ \sum_{i=1}^{n} X_i \right] \right\} \leq \frac{1}{k}.$$ 

(d) T F If all capacities in a flow network are integers, then the value of the maximum flow must also be an integer.
(e) T F If there is a polynomial-time reduction from the NP-hard problem A to another NP-hard problem B, and there is also an efficient $c$-approximation algorithm to B; then there is also an efficient $c$-approximation algorithm for A.

(f) T F The following is correct pseudocode for deleting an element $x \in \{0, 1, \ldots, u-1\}$ from a van Emde Boas tree $V$ in $O(lg \ lg u)$ time:

```
DELETE(V, x)
1   DELETE(V.cluster[high(x)], low(x))
2   if V.cluster[high(x)] is empty:
3       DELETE(V.summary, high(x))
```
(g) T F If we build a skip list on \( n \) elements, then we can search for any element in worst case \( O(\lg n) \) time.

(h) T F If a graph has a unique shortest path \( P \) from node \( s \) to node \( t \), and has a unique minimum spanning tree \( T \), then every edge in \( P \) must also be in \( T \).
Problem 2. Short Answer [48 points] (6 parts)

(a) Suppose that you are given a black-box algorithm that can solve linear programs in exactly $n^{3.5}$ milliseconds, where $n$ is the number of variables. Your boss gives you a “mixed-integer” linear program with 5 variables constrained to binary values, and $k$ variables that are only constrained to real numbers. How would you solve this problem efficiently, and about how much time would you expect it to take?

(b) Given a weighted directed graph $G = (V, E, w)$ and two nodes $s, t \in V$, write a linear program whose optimal value is the weight of the minimum $s$-$t$ cut, and justify why this is true.
(c) Write the dual of the following linear program:

\[
\begin{align*}
\text{maximize} & \quad 3x - 2y \\
\text{subject to} & \quad x + 5y \leq 4 \\
& \quad x, y \geq 0
\end{align*}
\]

(d) Consider a Markov Chain on the state space \( S = \{0, 1, \ldots, T\} \), where \( T \in \mathbb{N}^+ \). The transition probabilities are given by \( \Pr[i \to i + 1] = p < 0.5 \) with \( p > 0 \), and \( \Pr[i \to 0] = 1 - p \) for \( i \in \{0, 1, \ldots, T - 1\} \), and \( \Pr[T \to T] = 1 \). Find a stationary distribution of this Markov Chain\(^1\) and prove that it is stationary.

\(^1\)This Markov Chain has been nicknamed the "Tenure Game." With probability \( p \) you take one step towards tenure, or with probability \( 1 - p \) you lose it all. Once you have tenure, you keep it forever.
(e) Suppose you are given a nonnegatively weighted directed graph \( G = (V, E, w) \), and you have already computed the all-pairs shortest-path weights \( \delta(u, v) \) for all \( u, v \in V \). Now you add a weighted directed edge \((x, y)\), where \( x, y \in V \), to form a graph \( G' = (V, E', w') \) where \( E' = E \cup \{(x, y)\} \). Show how to (re)compute the all-pairs shortest-path weights \( \delta'(u, v) \) in \( G' \) using \( O(V^2) \) time.
(f) We can implement a queue using two stacks $S_e$ and $S_d$ as follows. To enqueue $x$ into the queue, we push $x$ onto $S_e$. To dequeue from the queue, we pop and return the top item from $S_d$. However, if $S_d$ is empty, we first fill it (and empty $S_e$) by popping the top item from $S_e$, pushing this item onto $S_d$, and repeating until $S_e$ is empty. (You do not need to prove that this data structure correctly implements a queue.) Assuming that push and pop operations take $O(1)$ worst-case time, prove that the enqueue and dequeue operations take $O(1)$ amortized time (when starting from an empty queue).
We can also use an accounting method by putting two coins on the enqueued item and using those coins to pay for the two pops during the deque.
Problem 3. Final Draft [15 points] (2 parts)

Your recent success as the General Manager of the Arizona Ordinals earned you quite a reputation. The team went undefeated and won the Uber Bowl. You’ve now been asked to consult on the drafting process for a basketball team, the Philadelphia 6.046ers. Unfortunately, the goals of the team, as well as the drafting process (which more closely resembles free agency) are different. Specifically, the owner doesn’t care about the skill level of his team. He just wants to build the biggest team possible. In addition, the players are much more fickle. Each player approaches the owner one at a time, announces the positions he’s willing to play, and leaves the meeting signed to a specific position, or unsigned at all, and never returns. It is still the case that every player may be assigned to at most one position, and every position can be filled by at most one player.

Edges of a bipartite graph $G = (L \cup R, E)$ are revealed to you in the following online manner: Each node $u \in L$ is revealed, one at a time, along with a linked list of every edge in $E$ incident to $u$. Upon revealing $u$, you must either:

1. Match $u$ to an unmatched neighbor of $u$ in $R$, and keep this edge forever.
2. Throw away $u$ and never match it.

After processing every node in $L$, you will have some matching, APX, with $|\text{APX}|$ edges. Let OPT denote the maximum-cardinality matching in $G$.

(a) [5 points] Show that no deterministic algorithm can guarantee $|\text{APX}| > |\text{OPT}|/2$.

Hint: Design two graphs $G$ and $G'$ (with the same $L$ and $R$) that are identical after only one node has been revealed, but would require you to choose a different edge. One possible example has $|L| = |R| = 2$. 
(b) [10 points] Design an efficient, deterministic algorithm that guarantees $|\text{APX}| \geq \frac{|\text{OPT}|}{2}$ for all bipartite graphs $G$, prove that it obtains this guarantee, and show that it runs in polynomial time.
Problem 4. Final Countdown [15 points]

You’re the DJ of 604.6, a cheesy 80s pop radio station. Unfortunately, cheesy 80s pop isn’t as popular as it used to be and you’re in need of some funding to stay afloat. In order to raise funds, you’ve decided to air a countdown of the best cheesy 80s pop one-hit wonders ever. This turns out to be a disastrous idea, as the only people who listen to cheesy 80s pop one-hit wonders are the artists themselves, and they’re all broke. Still, they like to hear their songs on the radio, and if they hear their own song, they’ll donate one dollar. Each artist truly loves listening to cheesy 80s pop, except for a few songs that they just can’t stand. Therefore, they’ll listen to the entire countdown until they hear a song they can’t stand, then they’ll stop. Your manager says you need to raise $k$ dollars or the station will go bankrupt. Your job is to decide whether there exists an ordering of cheesy 80s pop one-hit wonders so that at least $k$ artists will hear their own song.

In the POP problem, you are given an integer $k$, a set $V$ of $n$ artists, and for each artist $v \in V$, you are given the set $E(v)$ of other artists that $v$ hates. Your goal is to output an ordered list $L$ of all $n$ artists. Call an artist $v$ satisfied by $L$ if, in the $L$ ordering, $v$ comes before every artist that $v$ hates (i.e., every element of $E(v)$). The POP problem asks whether it is possible to satisfy at least $k$ artists by some ordered list $L$.

Prove that POP is NP-complete. You may find the following known NP-complete problem useful, although you are not required to use it:

**INDIE (Independent Set):** Given as input an undirected graph $G = (V, E)$ and an integer $k$, call a subset of nodes $S \subseteq V$ independent if there are no edges between any two nodes in $S$ (formally, $(u, v) \notin E$ for all $u, v \in S$). The Independent Set problem asks whether there exists an independent set of size at least $k$ in $G$. 
(continue your answer here, if needed)
Problem 5. Final Act [15 points]

As part of a traveling magic act, your final trick involves walking across a long stage covered in walls of fire and coming out unharmed! To the audience this looks quite impressive, but the secret is in where the flames lie. You always align the fire walls vertically or horizontally with the stage, and do so in a way so that no two walls touch (so there’s always a gap you can walk through without actually touching the fire, but the audience doesn’t notice this in the dimly lit room). Because you’re part of a traveling act, you have to set up a new stage every time. You’ve noticed that the audience gets more excited the more fire walls there are, so your goal is to get the most fire walls on stage at each venue. Each venue has a preset list of horizontal/vertical segments where they are capable of setting up fire walls.

You are given as input a set $V$ of (possibly intersecting) vertical segments and a set $H$ of (possibly intersecting) horizontal segments. (Two segments intersect if they share any point in the plane.) Design an efficient 2-approximation algorithm for finding the largest non-intersecting subset of $V \cup H$. (A set of segments is non-intersecting if no two segments in the set intersect.) Prove that your algorithm computes a solution $APX$ such that $|APX| \geq \frac{1}{2}|OPT|$, where $OPT$ denotes the largest non-intersecting subset of $V \cup H$.

Hint: How would you solve the problem in the special case where $V = \emptyset$?
Problem 6. Final Project [15 points]

Ben Bitdiddle is a senior in Course 6, and he wants to make a Photo Album webapp for his 6.UAP. He has already devised a storage method for photos that allows constant-time virtual references for individual photos on the server, and has a neat GUI for users to construct and look through their albums: all that is left is the nasty back-end work, i.e., making his GUI actually functional. Ben is looking for the most efficient way to handle the maintenance of these albums.

Design an efficient data structure that stores a set $S$ of objects, and maintains a consistent (but arbitrary) ordering $<$ of the objects. **Consistent** means that, once your data structure decides $o < o'$ for two objects $o, o' \in S$, it will never later decide that $o' < o$. Your data structure must support the following operations:

- **INSERT(o)**: Add object $o$ to the structure. Assume that $o \notin S$.
- **DELETE(o)**: Remove object $o$ from the structure, if $o \in S$.
- **NEXT(o)**: Move to the next object in the structure after $o \in S$.
- **PREVIOUS(o)**: Move to the previous object in the structure before $o \in S$.
- **SEARCH(o)**: Determine whether $o \in S$.

Describe your structure, and give **constant-time** algorithms to support these operations. State which, if any, of your time bounds are amortized and/or expected. (Partial credit will be given for slower solutions.)
Problem 7. Final Destination(s) [20 points] (3 parts)

You are trying to decide whether two horror movies have identical story lines. Fortunately, each story line has a simple tree-like structure: there is a distinct starting point and the plot diverges along separate paths at various later points. Two story lines are deemed identical if their plots diverge in the same ways after their respective starting points.

For two rooted, unordered trees $T_1$ and $T_2$ each on $n$ nodes, call $T_1$ and $T_2$ interchangeable if there exists a one-to-one correspondence $f$ from the nodes of $T_1$ to those of $T_2$ such that

1. the root $r_1$ of $T_1$ is mapped to the root $r_2$ of $T_2$;
2. $v$ is a child of $w$ in $T_1$ if and only if $f(v)$ is a child of $f(w)$ in $T_2$.

Associate a polynomial $P_v$ with each vertex $v$ recursively as follows. At the base case, a leaf vertex $v$ has associated polynomial $P_v = x_0$. For an internal vertex $v$ of height $h$ having children $v_1, v_2, \ldots, v_k$, its associated polynomial is

$$P_v = (x_h - P_{v_1})(x_h - P_{v_2})\cdots(x_h - P_{v_k}).$$

Note that there is exactly one variable for each level in the tree.

(a) [8 points] Prove that the degree of $P_{r_i}$ is at most $n$. (The degree of a polynomial is the maximum degree of its terms. The degree of a term $x_1^{\alpha_1}x_2^{\alpha_2}\cdots x_n^{\alpha_n}$ is $\alpha_1 + \alpha_2 + \cdots + \alpha_n$. For example, the degree of $x_1^2x_2 + x_2x_3^3$ is $\max\{2 + 1, 1 + 3\} = 4$.)
(b) [8 points] Prove that $P_{r_1} = P_{r_2}$ (for all values of $x_1, x_2, \ldots$) if and only if $T_1$ and $T_2$ are interchangeable.
(c) [4 points] Give a linear-time algorithm for testing whether two given rooted trees are interchangeable. Your algorithm should be correct with probability at least $\frac{2}{3}$.

*Hint:* Assume (a) and (b), and use the following theorem of Schwartz and Zippel.

**Theorem:** Let $P$ be a non-zero multivariate polynomial in $x_1, x_2, \ldots, x_n$ of degree $d$. If $r_1, r_2, \ldots, r_n$ are chosen uniformly at random from $\{1, 2, \ldots, m\}$, then

$$\Pr \left\{ P(r_1, r_2, \ldots, r_n) = 0 \right\} \leq \frac{d}{m}.$$
Problem 8. Final Boss [15 points]

After many hours of playing Xelda: Brilliant Blade, the latest installment in the series, you have finally made it to the final boss battle with the evil wizard Cannon. His signature move involves climbing onto a chandelier attached to the ceiling and diving at you. Since the chandelier is very high above the ground, this attack can be very dangerous if it connects. Your first priority, therefore, is to ensure that he cannot perform this move.

You notice that the chandelier is attached to the ceiling by a complex network of enchanted chains. The chains are impenetrable, but you notice that whoever put the chains together doesn’t know how to weld properly. Wherever two or more chains meet up, or wherever a chain is connected to the chandelier or the ceiling, they are connected by a weak, brittle welding joint. You figure that if you slash your sword through one of these joints, the joint will shatter, effectively disconnecting all chains that were attached to it.

However, not all joints have the same brittleness. Your companion, Phi, can use her magic powers to determine how easy it would be to break through each joint. Armed with this knowledge, you set out to dislodge the chandelier from the ceiling while minimizing the amount of work you need to do.

You are given a directed graph \( G = (V, E) \), two disjoint subsets \( A, B \subseteq V \), and a vertex weight function \( w : V \to \mathbb{R}^+ \). Call a subset of vertices \( S \subseteq V \) winning if the removal of \( S \) disconnects all vertices in \( A \) from all vertices in \( B \), i.e., every path in \( G \) from a vertex in \( A \) to a vertex in \( B \) must contain at least one vertex in \( S \). Give a polynomial-time algorithm to find a winning set of vertices of minimum total weight.

(Note that \( A \cup B \) is not necessarily \( V \), and that \( S \) may contain vertices that are in \( A \) or \( B \).)
(continue your answer here, if needed)