Final Examination

- Do not open this exam booklet until you are directed to do so.
- This exam ends at 4:30 P.M. It contains 8 problems, some with several parts. You have 180 minutes to earn 160 points.
- This exam is closed book, but you may use two double-sided 8 1/2" × 11" or A4 crib sheets.
- When the exam begins, write your name in the space below and on the top of every page in this exam. Circle your recitation instructor.
- Write your solutions in the space provided. If you need more space, use the scratch paper at the end of the exam booklet. Please write your name on any extra pages that you use.
- **Do not spend too much time on any one problem.** Read them all through first, and attack them in the order that allows you to make the most progress.
- Do not waste time rederiving algorithms and facts that we have studied. It suffices to cite known results.
- Show your work, as partial credit will be given. You will be graded on the correctness and efficiency of your answers and also on your clarity. Please be neat.
- When giving an algorithm, sketch a proof of its correctness and analyze its running time using an appropriate measure.
- Good luck!

<table>
<thead>
<tr>
<th>Problem</th>
<th>Points</th>
<th>Grade</th>
<th>Initials</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>48</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Problem</th>
<th>Points</th>
<th>Grade</th>
<th>Initials</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>19</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Total   | 160    |       |          |

Name: ____________________________________________

Circle your recitation instructor:

TB (F10, F11)    Ammar (F12,F1)   Angelina (F2,F3)
Problem 1. True/False and Justify [48 points] (12 parts)
Circle T or F for each of the following statements to indicate whether the statement is true or false, respectively. If the statement is correct, briefly state why. If the statement is wrong, explain why. Your justification is worth more than your true or false designation.

(a) T F [4 points] If a sequence of $n$ operations on a data structure cost $T(n)$, then the amortized runtime of each operation in this sequence is $T(n)/n$.

(b) T F [4 points] Fix some integers $m >> n > 0$. For every function $h : [m] \rightarrow [n]$ in a universal hashing family $\mathcal{H}$, there exists an integer $0 \leq i \leq m - 1$ such that $h(i) \neq 0$. 
(c) T F [4 points] If a problem in NP can be solved in polynomial time, then it is known that all problems in NP can be solved in polynomial time.

(d) T F [4 points] If P = NP, then every nontrivial decision problem L ∈ P is NP-complete. (A decision problem L is nontrivial if there exist some x, y such that x ∈ L and y \notin L.)
(e) T F  [4 points] A spanning tree of a given undirected, connected graph $G = (V, E)$ can be found in $O(E)$ time.

(f) T F  [4 points] Consider the following algorithm for computing the square root of an $n$-bit integer $x$:

\begin{verbatim}
SQUARE-ROOT(x)
  For $i = 1, 2, \ldots, \lfloor x/2 \rfloor$:
    If $i^2 = x$, then output $i$.
\end{verbatim}

This algorithm runs in polynomial time.
(g) T F [4 points] If all edge capacities in a flow network are integer multiples of 3, then the maximum flow value is a multiple of 3.

(h) T F [4 points] Given a connected directed graph $G = (V, E)$ and a source vertex $s \in V$ such that each every $e \in E$ has an integer weight $w(e) \in \{0, 1, \ldots, V^3\}$, there is an algorithm to compute single-source shortest-path weights $\delta(s, v)$ for all $v \in V$ in $O(E \lg \lg V)$ time.
(i) T F [4 points] Given a constant \( \varepsilon > 0 \), probabilistic property testing whether a sequence is \( \varepsilon \)-close (as defined in the lecture) to monotone requires \( \Omega(n) \) queries.

(j) T F There is a sublinear-time algorithm that decides whether a given undirected graph is connected.

(k) T F [4 points] An adversary can force a skip-list insertion to take \( \Omega(n) \) time.
(I) T F [4 points] Assume $P \neq NP$. The Traveling Salesman Problem has a polynomial-time $\alpha$-approximation algorithm for some constant $\alpha > 1$. 
Problem 2. **Short answer** [33 points] (5 parts)

Give brief answers to the following problems.

(a) [9 points] Match up each application with an algorithm or data structure that we used to solve it in this course. Use each answer exactly once.

- Map folding
- Integer multiplication
- Finding a minimum spanning tree
- All-pairs shortest paths
- Polynomial identity testing
- SubsetSum is NP-hard
- Chinese postman tour
- Finding a minimum cut
- Single-source shortest paths

A. Polynomial reduction
B. Ford-Fulkerson algorithm
C. Dynamic programming
D. General matching
E. Divide and conquer
F. Johnson algorithm
G. Dijkstra
H. Greedy algorithm
I. Monte Carlo algorithm
(b) [6 points] Suppose that you are given an unsorted array $A$ of $n$ integers, some of which may be duplicates. Explain how you could “uniquify” the array (that is, output another array containing each unique element of $A$ exactly once) in $O(n)$ expected time.

(c) [6 points] Prove that there is no polynomial-time $(1 + \frac{1}{2n})$-approximation algorithm for Vertex Cover (unless $P = NP$).
(d) [6 points] The following table gives the frequencies of the characters of an alphabet.

<table>
<thead>
<tr>
<th>Character</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1/20</td>
</tr>
<tr>
<td>B</td>
<td>2/20</td>
</tr>
<tr>
<td>C</td>
<td>2/20</td>
</tr>
<tr>
<td>D</td>
<td>4/20</td>
</tr>
<tr>
<td>E</td>
<td>4/20</td>
</tr>
<tr>
<td>F</td>
<td>7/20</td>
</tr>
</tbody>
</table>

Show a tree that Huffman’s algorithm could produce for these characters and frequencies, and fill in the table below with the codeword for each character in the alphabet produced by this tree.

<table>
<thead>
<tr>
<th>Character</th>
<th>Codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
</tr>
</tbody>
</table>
(e) [6 points] A nonvertical line $L$ in the plane can be represented by an equation $y = m_L x + b_L$ for real numbers $m_L, b_L$. A point $P = (x_P, y_P)$ is above a line $L$ if $y_P \geq m_L x_P + b_L$.

Given $n$ nonvertical lines $L_1, L_2, \ldots, L_n$ in the plane, describe how to find in linear time the point $P$ of minimum $y$ coordinate that is above all $n$ lines.
Problem 3. Hadamard chronicles IV: Divide and conquer [15 points]

For each nonnegative integer \( k \), the \textbf{Hadamard matrix} \( H_k \) is the \( 2^k \times 2^k \) matrix defined recursively as follows:

- \( H_0 = [1] \).
- For \( k > 0 \), \( H_k = \begin{bmatrix} \frac{H_{k-1}}{H_{k-1}} & H_{k-1} \\ \frac{-1}{H_{k-1}} & -H_{k-1} \end{bmatrix} \).

Let \( \vec{v} \) be a column vector of length \( n = 2^k \). Describe an algorithm that computes the product \( H_k \vec{v} \) in \( O(n \log n) \) arithmetic operations (additions, subtractions, multiplications, or divisions). Show that your algorithm achieves the stated complexity.

Given an undirected graph $G = (V, E)$ and two vertices $s, t \in V$, a maximum $s$-$t$ cut is a cut $(S, T)$ satisfying the following conditions:

i. $(S, T)$ is a cut: $S, T \subset V$, $S \cap T = \emptyset$, and $S \cup T = V$.

ii. $s \in S$ and $t \in T$.

iii. The number of edges $(u, v) \in E$ with $u \in S$ and $v \in V \setminus S$ is the maximum possible.

The MAXIMUM-$s$-$t$-CUT problem is to find a maximum cut for a given pair of vertices. Unlike its counterpart, the MINIMUM-$s$-$t$-CUT problem, MAXIMUM-$s$-$t$-CUT is NP-hard. Analyze the following algorithm and show that it is a 2-approximation algorithm for MAXIMUM-$s$-$t$-CUT.

MAX-CUT($G, s, t$)
1. $S \leftarrow \{s\}$
2. $T \leftarrow \{t\}$
3. for each vertex $v \in V - \{s, t\}$
4.     do
5.         $a \leftarrow$ the number of edges $(u, v)$ with $u \in S$.
6.         $b \leftarrow$ the number of edges $(v, w)$ with $w \in T$.
7.         if $a > b$
8.             then
9.                 $T \leftarrow T \cup \{v\}$
10.            else
11.                $S \leftarrow S \cup \{v\}$
12.        return $(S, T)$ as the approximation for the MAXIMUM-$s$-$t$-CUT of $s$ and $t$. 
Problem 5. Be the computer [10 points]
Starting from the following flow (printed above or to the right of the capacities), perform one iteration of the Edmonds-Karp algorithm.

(a) [4 points] Write down your shortest augmenting path, that is, the augmenting path with the fewest possible edges.
(b) [3 points] Perform the augmentation. What is the value of the resulting flow?

(c) [3 points] Is the resulting flow optimal? If so, give a cut whose capacity is equal to the value of the flow. If not, give a shortest augmenting path.
Problem 6. Graphs and paths and cycles, oh my! [15 points]

Given a directed graph $G = (V, E)$, a Hamiltonian path is a path that visits each vertex in $G$ exactly once. Consider the following properties for a directed graph $G$:

- $P_1(G)$: $G$ contains either a cycle (not necessarily Hamiltonian) or a Hamiltonian path (or both).
- $P_2(G)$: $G$ contains both a cycle (not necessarily Hamiltonian) and a Hamiltonian path.

Given that the problem HAM-PATH (which decides whether a graph $G$ has a Hamiltonian path) is NP-complete, prove that one of the two properties above is decidable in polynomial time, while the other property is NP-complete.
Problem 7.  **If I only had a black box...**  [10 points]

Suppose you are given a magic black box that, in polynomial time, determines the number of vertices in the largest complete subgraph of a given undirected graph $G$. Describe and analyze a polynomial-time algorithm that, given an undirected graph $G$, computes a complete subgraph of $G$ of maximum size, using this magic black box as a subroutine.
Problem 8. Hero Training [19 points]

You are training for the World Championship of Guitar Hero World Tour, whose first prize is a real guitar. You decide to use algorithms to find the optimal way to place your fingers on the keys of the guitar controller to maximize the ease by which you can play the 86 songs.

Formally, a note is an element of \{A, B, C, D, E\} (representing the green, red, yellow, blue, and orange keys on the guitar). A chord is a nonempty set of notes, that is, a nonempty subset of \{A, B, C, D, E\}. A song is a sequence of chords: \(c_1, c_2, \ldots, c_n\). A pose is a function from \{1, 2, 3, 4\} to \{A, B, C, D, E, \emptyset\}, that is, a mapping of each finger on your left hand (excluding thumb) either to a note or to the special value \(\emptyset\) meaning that the finger is not on a key. A fingering for a song \(c_1, c_2, \ldots, c_n\) is a sequence of \(n\) poses \(p_1, p_2, \ldots, p_n\) such that pose \(p_i\) places exactly one finger on each note in \(c_i\), for all \(1 \leq i \leq n\).

You have carefully defined a real number \(D[p, q]\) measuring the difficulty of transitioning your fingers from pose \(p\) to \(q\), for all poses \(p\) and \(q\). The difficulty of a fingering \(p_1, p_2, \ldots, p_n\) is the sum \(\sum_{i=2}^{n} D[p_{i-1}, p_i]\). Give an \(O(n)\)-time algorithm that, given a song \(c_1, c_2, \ldots, c_n\), finds a fingering \(p_1, p_2, \ldots, p_n\) of the song with minimum possible difficulty.