CSE 101 Final Exam

Winter 2015

Instructions: Do not open until the exam starts. The exam will run for 180 minutes. The problems are roughly sorted in increasing order of difficulty. Answer all questions completely. You are free to make use of any result in the textbook or proved in class. You may use up to 12 1-sided pages of notes, and may not use the textbook nor any electronic aids. Write your solutions in the space provided, the pages at the end of this handout, or on the scratch paper provided (be sure to label it with your name). If you have solutions written anywhere other than the provided space be sure to indicate where they are to be found.

For several problems on this exam an algorithm is asked for with a specific runtime. Providing a working algorithm with worse runtime will usually receive some partial credit.

Name:

ID Number:

<table>
<thead>
<tr>
<th>Problem</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Score</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Question 1 (Minimum Spanning Tree, 15 points). Compute the minimum spanning tree of the following graph.
Question 2 (Backtracking, 15 points). Irene is trying to show that the graph below does not have a Hamiltonian cycle. Use backtracking to show that none exists.
Question 3 (Lilypad Hopping, 15 points). Jeremiah was a bullfrog. He lives in a pond with lilypads located at coordinates \((x_i, y_i)\) for \(1 \leq i \leq n\). Jeremiah can jump between two lilypads if they are separated by a distance of at most 1 unit. Give an algorithm that with inputs \((x_i, y_i)\) determines whether or not it is possible for Jeremiah to reach the lilypad at location \((x_n, y_n)\) from the one at location \((x_1, y_1)\) by a sequence of such hops, and analyze the runtime. For full credit, your algorithm should run in time \(O(n^2)\).
**Question 4** (Longest Palindromic Subsequence, 15 points). Consider the problem of finding the longest palindromic subsequence of a sequence $x_1, x_2, \ldots, x_n$. In particular, this means to find the longest subsequence $y_1, y_2, \ldots, y_k$ (where $y_i = x_{a_i}$ for some sequence $a_1 < a_2 < \ldots < a_k$) so that $y_1 = y_k, y_2 = y_{k-1}, \ldots, y_i = y_{k+1-i}, \ldots, y_k = y_1$.

Give an algorithm to find the length of the longest palindromic subsequence and prove its correctness. For full credit, your algorithm should have runtime $O(n^2)$. 

Question 5 (Local Maximum Search, 20 points). Given a sequence of numbers $a_1, a_2, \ldots, a_n$, we say that $a_i$ is a local maximum if $a_i \geq a_{i+1}, a_{i-1}$ (where the condition is ignored if $a_{i+1}$ would be out of range). Give an algorithm that given a sequence $a_1, a_2, \ldots, a_n$ finds a local maximum and analyze its runtime. For full credit, your algorithm should run in time $O(\log(n))$.

Hint: Divide the array in two and figure out a way to select one half that is guaranteed to have a local maximum.
Question 6 (Approximation Algorithm for Bin Packing, 20 points). Consider the following problem. Given \(n\) items of weights \(w_1, w_2, \ldots, w_n \leq 1\) you are tasked with the problem of putting these items into bins so that the total weight of all items in any one bin is at most 1. So for example, if you had items of weight 0.6, 0.5 and 0.3, you could put them all in different bins, or put the first and third in a bin together and the second in a different bin, but you could not put the first and second in the same bin because 0.6 + 0.5 > 1.

Your objective is to do this while minimizing the total number of bins used. It turns out that this problem is NP-Hard (there is a relatively simple reduction to Knapsack).

Give a polynomial time algorithm that provides a 2-approximation for this problem, and prove correctness.

Hint: As long as your bins are full enough that you cannot combine any pair of them, the average weight in each bin is at least 1/2.