Instructions: Do not open until the exam starts. The exam will run for 45 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 6 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.

Name:

--------------------------------------------------------------------------------------

ID Number:

--------------------------------------------------------------------------------------

<table>
<thead>
<tr>
<th>Problem</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Score</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Question 1 (Computing Runtime, 30 points). Consider the following sorting algorithm:

\textbf{RecursiveSort}(A, i, j) \ \ \text{// sorts the elements of A between indices i and j}
\begin{align*}
    &\text{if } j = i + 1 \\
    &\quad \text{if } A[i] > A[j] \\
    &\quad \quad \text{swap}(A[i], A[j]) \\
    &\quad \text{else} \\
    &\quad \quad \text{Set } k = \text{ceiling}(2\times(j-i+1)/3) - 1 \\
    &\quad \quad \text{RecursiveSort}(A, i, i+k) \\
    &\quad \quad \text{RecursiveSort}(A, j-k, j) \\
    &\quad \quad \text{RecursiveSort}(A, i, i+k)
\end{align*}

(a) Give a recurrence for the runtime of this algorithm on an array of length \( n \) (when \( i = 1 \) and \( j = n \)).

(b) What is the asymptotic runtime of this algorithm (as a big-\( \Theta \))?
**Question 2** (Longest Root to Leaf Path, 35 points). Let $T$ be a balanced binary tree on $n$ vertices (so $n$ is one less than a power of 2) with (possibly negative) edge weights. Give an $O(n)$ time divide and conquer algorithm for computing the length of the longest root-to-leaf path in $T$. 
Question 3 (Shortest Paths From Path Lengths, 35 points). Suppose that you are given a graph $G$ with (possibly negative) edge weights. Additionally, you are given a vertex, $s$ and the lengths, $d(w)$ of the shortest path from $s$ to each other vertex $w$ (you may assume that shortest paths actually exist). Give a linear time algorithm that given another vertex $t$ returns the short path from $s$ to $t$ (the actual path and not just its length). You may assume that the shortest path is unique.