7200. Analysis of Algorithms

Midterm Exam

October 21, 2014

Id: .................................................................

• Answer both questions.
• Answer a question only within the given space by using a readable normal size font text.
• You get 20% of the credit if you leave the answer blank. You get no credit for a wrong answer.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Maximum Points</th>
<th>Your Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

Good Luck!
1. Assume \( n = 2^k \) is a power of 2. Let \( A = A[1] < A[2], \ldots < A[n] \) be a sorted array of \( n \) distinct positive integers. Let \( x \) be a positive even integer. Describe an efficient algorithm that finds an index \( i \) such that \( x/2 \leq A[i] \leq 2x \) if such an index exists. That is, the search is for any number in the array that is between \( x/2 \) and \( 2x \).

What is the exact worst-case number of comparisons made by your algorithm? Justify the correctness and complexity of your algorithm.
If you are not sure about your previous algorithm, describe any (possibly not efficient) correct algorithm that solves the problem. What is the worst-case number of comparisons made by your algorithm? Justify the correctness and complexity of your algorithm.
2. Let $A = A[1], A[2], \ldots, A[n]$ be an unsorted array of $n$ distinct integers. For any index $i$ between 1 and $n$, define $B[i]$ to be the smallest index $j$ such that $A[j] > A[i]$ and $i < j \leq n$, if such an index exists. Otherwise, $B[i] = n + 1$. Design an efficient algorithm that constructs the array $B$ for a given array $A$.

What is the worst-case number of comparisons made by your algorithm? Justify the correctness and complexity of your algorithm.
If you are not sure about your previous algorithm, describe any (possibly not efficient) correct algorithm that solves the problem. What is the worst-case number of comparisons made by your algorithm? Justify the correctness and complexity of your algorithm.
3. Let $A$ and $B$ be two unsorted arrays each contains $n$ positive integers. Assume that all $2n$ integers are distinct. We say that $A$ is **totally greater than** $B$ if it is possible to re-order both arrays in a way that $A[i] > B[i]$ for every index $1 \leq i \leq n$. Describe an efficient algorithm that for given arrays $A$ and $B$ determines if $A$ is totally greater than $B$.

What is the worst-case number of comparisons made by your algorithm? Justify the correctness and complexity of your algorithm.
If you are not sure about your previous algorithm, describe any (possibly not efficient) correct algorithm that solves the problem. What is the worst-case number of comparisons made by your algorithm? Justify the correctness and complexity of your algorithm.