CS140 - Midterm 1: Sample

1. **[7.5 points]** T/F - State whether the statements below are true or false **AND** give a *brief* justification for your answer.

   _______ $2^{c\sqrt{n}} = O(2^{\sqrt{n}})$ for any constant $c > 0$

   _______ $f(n)+g(n) = O(\max(f(n), g(n))$ assuming $f(n)$ and $g(n)$ are positive functions.

   _______ You are given two algorithms A_1 and A_2 for solving a problem. A_1 runs in time $O(n^3)$ and A_2 runs in time $O(\log n)$. It is possible for A_1 to take less time to run than A_2 on all possible inputs.

   _______ A k-sorted array is an array where any value is no more than k positions from it's correct location. The worst case running time of Insertion-Sort on a k-sorted array is $O(n^2)$.

   _______ If f is $O(g)$, then 2^f is $O(2^g)$

2. [**6 points**] You're given an array of n elements and would like to print the k largest in sorted, *decreasing* order. For example, if $n = 8$ and $k = 3$ and the input were:

```
8 10 2 1 4 6 2 15
```

Then the output would be: `15 10 8`

For each of the methods below, describe the *most efficient*, worst-case run-time for the method described. Note your run-times should be in terms of n **and** k.

(a) Sort all n numbers and then print the largest k.

(b) Find the largest value. Remove it from the array and print it. Repeat until you've found the k largest values.

(c) Find the kth largest number, partition around it, then sort the k largest numbers.

3. [**6 points**] Suppose you are given an array $A[1...n]$ of sorted integers that has been rotated k positions to the right. For example, $[35, 42, 5, 15, 27, 29]$ is a sorted array that has been circularly rotated $k = 2$ positions, while $[27, 29, 35, 42, 5, 15]$ has been rotated $k = 4$ positions. Describe an algorithm to find the largest value in a k-shifted array in $O(\log n)$ time.

4. **[6 points]** If possible, solve the following recurrences and prove that your answer is correct (using the master method is fine as proof):

   (a) $T(n) = 3T(\frac{n}{3}) + \log n$

   (b) $T(n) = T(n-1) + n^d \log n$, for $d \geq 1$