Practice Midterm 1 (answers on the back)

1. Indicate whether the following statements are True or False. Assume n, $f()$, $g()$, and $h()$ are positive and n is an integer You do not need to give a proof.

   _____ $n^2 \in O(n^3)$.

   _____ If $f(n) \in O(g(n))$ and $g(n) \in O(h(n))$ then $f(n) \in O(h(n))$.

   _____ If $f(n) \in O(g(n))$ and $g(n) \in O(h(n))$ then $f(n) + g(n) \in O(h(n))$.

   _____ $f(n) \in O(g(n))$ if and only if $g(n) \in \Omega(f(n))$.

   _____ $\sqrt{n} \in O(n^{\sin n})$ where $\sin n$ means sin of n degrees.

2. Describe a $\Theta(n \log n)$-time algorithm that, given a set S of n real numbers and another real number x, determines whether or not there exist two elements of S whose sum is exactly x.

3. Suppose that you are given n red and n blue water jugs. For every red jug there is a blue jug that holds the same amount of water, and vice versa. Your task is to pair-up each red jug with the blue jug that holds the same amount of water. The only operation you may perform is to pick a red jug and a blue jug, fill the red jug with water, and then pour the water into the blue jug. This will tell you if the red or the blue jug can hold more water, or if they can hold the same amount. Note: You may not directly compare two red jugs or two blue jugs.

   Prove that the number of operations you must perform to pair up the jugs is $\Omega(n \log n)$.

4. Use a recursion tree to give an asympotically tight solution to the recurrence $T(n) = T(\alpha n) + T((1-\alpha)n) + cn$, where α is a constant in the range $0 < \alpha < 1$ and $c > 0$ is also a constant.

Solution Sketches

1. True. $n^2 \leq n^3 \iff 1 \leq n$.

   True. $f(n) \leq cg(n)$ (for all $n \geq n_0$) and $g(n) \leq dh(n)$ (for all $n \geq n_1$) implies $f(n) \leq cdh(n)$ (for all $n \geq \max\{n_0, n_1\}$).

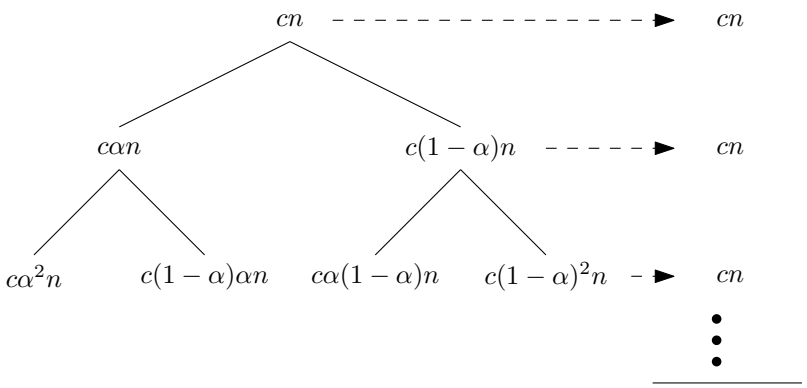
   True. $f(n) \leq cg(n)$ (for all $n \geq n_0$) and $g(n) \leq dh(n)$ (for all $n \geq n_1$) implies $f(n) + g(n) \leq (cd + d)h(n)$ (for all $n \geq \max\{n_0, n_1\}$).

   True. $f(n) \leq cg(n)$ (for all $n \geq n_0$) implies $g(n) \geq \frac{1}{c} f(n)$ (for all $n \geq n_0$). Since $c > 0$, $1/c > 0$.

   False. If n is a multiple of 180 degrees $\sin n = 0$, so $\sqrt{n} > cn^{\sin n}$ for any constant c for all $n = 180k > c^2$.

2. Sort S to obtain a sorted list of its elements $s_1 \leq s_2 \leq \cdots \leq s_n$. Consider $s_1 + s_n$. If $s_1 + s_n < x$ then s_1 can't be one of the two numbers whose sum is x because even adding the largest number (s_n) to it results in a number smaller than x. Similarly, if $s_1 + s_n > x$ then s_n can't be one of the two numbers whose sum is x. In either case, we eliminate one number from consideration and are left with the identical problem with one fewer number. We used constant time to eliminate one number, so the time, after sorting, to find the two numbers whose sum is x (or to eliminate all pairs) is $O(n)$. The total time of the algorithm is thus dominated by the time to sort which is $O(n \log n)$.

3. We can model any algorithm using a decision tree in which each internal node represents a comparison of a red and blue jug that has 3 outcomes (red $<$ blue, red $=$ blue, red $>$ blue). Since no other operations are permitted, the outcome of a sequence of such comparisons must determine a unique pairing of red and blue jugs. So every leaf of the decision tree represents at most one possible pairing. There are $n!$ possible pairings that the algorithm must be able to distinguish and thus the decision tree has at least $n!$ leaves. Since the number of leaves is at most 3^{depth}, the depth of the decision tree (which is the worst case number of comparisons you perform) is at least $\log_3(n!) \in \Omega(n \log n)$.

4. 

   $\sum_{i=0}^{\log_A n} cn \leq T(n) \leq \sum_{i=0}^{\log_B n} cn$

   Here $A = \max\{1/\alpha, 1/(1-\alpha)\}$ and $B = \min\{1/\alpha, 1/(1-\alpha)\}$. The leaf costs are at most a constant times the internal node costs, so $T(n) \in \Theta(n \log n)$.