CSE 3500 Algorithms and Complexity
Exam I, October 20, 2009

Note: You are supposed to give proofs to the time bounds of your algorithms. Read the questions carefully before attempting to solve them.

1. Prove or disprove:
   - (8 points) $5n^{2.5} + 100n^2(\log n)^2 + 10^{1000} = \Theta(n^{2.5})$.
   - (8 points) $n^n = \Theta(2^{2n \log n})$. 
2. (16 points) What is the run time of the following algorithm?:

Algorithm Test\((n)\)
\[\text{for } i := 1 \text{ to } 3n \text{ do} \]
\[\quad \text{for } j := 1 \text{ to } 5i^2 + 4i \text{ do} \]
\[\quad \quad \text{Flip a 2-sided coin;}\]
3. (18 points) Input is an array \( a[1 : n] \) of arbitrary real numbers. The array could only be of one of the following two types: 1) **Type I:** All the elements in the array are distinct; or 2) **Type II:** The array has \( n^{2/3} \) copies of one element, the other elements being distinct. Present a Monte Carlo algorithm that determines the type of the array in \( O(n^{2/3} \log n) \) time. Show that the output of your algorithm will be correct with high probability.
4. (16 points) A department has to keep records of its employees such that the following operations can be performed on the records:

- **Find_Name**(SSN): Return the name of the person whose social security number is SSN; and
- **Find_SSN**(Name): Return the social security number of the person whose name is Name.

Present a data structure for keeping the records that will take \(O(\log n)\) time to perform each of the above operations, \(n\) being the number of persons in the department. You can use \(O(n)\) space.
5. (16 points) Show how you’d use Heapify to form a max-heap out of the following elements: 23, 12, 5, 6, 11, 17, 14, 8, 2, 34, 21. Show the tree that results after each application of Heapify.
6. (18 points) A and B are two different divide-and-conquer algorithms for solving the same problem P. A reduces a problem instance of size n into 16 subproblems each of size \(n/2\) and takes \(n^3\) time for the partition and combine steps. B reduces any problem instance of size n into 64 subproblems each of size \(n/8\) and takes \(n^2\) time for the partition and combine steps. Which algorithm would you use to solve P? Why?