CS 660, HOMEWORK 1 (DUE: SUNDAY SEPTEMBER 15, 11:59 PM)

INSTRUCTOR: HOA VU

Each question is worth 20 points. The extra credit question is worth 10 points.

When you are asked to design an algorithm, do the following: a) describe the algorithm, b) explain (or more rigorously prove) why it is correct, and c) provide the running time.

Unless specified otherwise, the problems are from the textbook by Jeff Erickson that we use.

(1) **Question 1:**
   - Solve $T(n) = T(n-1) + n^2$ using the recurrence tree method. In particular, what is big-O of $T(n)$. Hint: use the formula $\sum_{i=1}^{n} i^2 = n(n+1)(2n+1)/6$.
   - Solve $T(n) = 4T(n-1) + 99$ using the recurrence tree method.
   - Solve $T(n) = T(n/6) + T(n/20) + O(n)$ using the recurrence tree method.
   - Show that $2^{2n} \neq O(2^n)$.
   - Problem 6 (page 49): Solve B, E, H using master theorem.

(2) **Question 2:** Problem 37 (page 64).

(3) **Question 3:** Problem 21, part a (page 55).

(4) **Question 4:** The majority element is the element that occurs more than half of the size of the array ($\lceil (n/2) \rceil$). Explain how to find the majority element in an array of n numbers in $O(n)$ time. If there is no majority element, the algorithm should also report "no majority element". Hint: there is a two line solution based on what we covered in class.

(5) **Question 5:** Problem 3a (page 124)

(6) **Extra credit:** Problem 21, part b (page 55) or Problem 3b (page 124) . If you do both, you will get 20 extra points.