Assignment 1

Due: January 16, 2020

Your solutions must be typed (preferably typeset in LaTeX) and submitted as a hard-copy at the beginning of class on the day its due. When asked to provide an algorithm you need to give well formatted pseudocode, a description of how your code solves the problem, and a brief argument of its correctness.

Problem 1: Asymptotic Analysis Practice

(a) [5 points] Prove or disprove that $\log_k n \in O(\lg n)$ for any $k > 1$. (Note that lg refers to $\log_2$)

(b) [5 points] The following recurrence relation solves to $O(n \lg^2 n)$. Prove this by substition. Do not use the Master method.

$$T(n) = 2T\left(\frac{n}{2}\right) + n \lg n$$
$$T(1) = 0$$

(c) [5 points] Suppose that $f(n)$ and $g(n)$ are non-negative functions. Prove or disprove the following: if $f(n) \in O(g(n))$ then $2^{f(n)} \in O(2^{g(n)})$.

Problem 2: Peak-finding Given a set of real numbers stored in an array A find the index of a *Peak*, where a *Peak* is defined as an element that is larger or equal to the both the elements on its sides. (Note: you only need to return *a* peak, not the highest one.)

Example array:

-2	6	-1	4	9	-5	5

Assuming the array one based it has peaks at indices $\{2, 5, 7\}$.

(a) [5 points] Give a linear time algorithm to solve this problem. (This should be obvious)

(b) [10 points] Give a $O(\log n)$ time algorithm to solve this problem.

(c) [10 points] What if instead of a simple array you are given a square matrix, where a *Peak* is now defined as an element larger or equal to its four neighbours. Give a $O(n \log n)$ solution to this variant of the problem.