COMP 550.002: Fall 2023
Assignment 2

Announced: September 20, 2023

Due Date: October 4, 2023

All problems are collaborative. You can form a group of at most four students to collaborate. You MUST mention the names of your collaborators, and cite any material you took help from (including discussions on canvas by other students) except the textbook.

CLRS refers Cormen et al. textbook.

Problem 1 (30 Points)

For each of the following pair of functions $f(n)$ and $g(n)$, determine whether $f(n) = O(g(n))$, $f(n) = \Omega(g(n))$, $f(n) = \Theta(n)$, $f(n) = o(g(n))$, or $f(n) = \omega(g(n))$. You should indicate all asymptotic notations that apply.

	$f(n)$	$g(n)$
(a)	$10^{10}n^2 + 5n$	$n^3 + 1$
(b)	3^n	$n \cdot 2^n$
(c)	$n \lg n$	$10n \lg 10n$
(d)	$n^2 \lg n$	$n(\log_{10} n)^2$

Problem 2 (24 points)

[This problem is based on CLRS problem 3-3(a)] Order the following list of functions by O-notation. Thus, determine an arrangement $g_1(n), g_2(n), \ldots$ of these functions so that $g_1(n) = \Omega(g_2(n))$, $g_2(n) = \Omega(g_3(n)), \ldots$ (In other words, $g_2(n) = O(g_1(n))$, $g_3(n) = O(g_2(n)), \ldots$). Partition your list into equivalence classes such that functions $f(n)$ and $g(n)$ are in the same class if and only of $f(n) = \Theta(g(n))$.

n^2	$n!$	$n \lg n$	$(3/2)^n$
2^{2^n}	$n^{1/\lg n}$	1	$n \lg n$
$4^{\lg n}$	n	$(\lg n)^2$	2^n

To get partial credit, include your justification for each relation between and within equivalence classes, i.e., state why $g_1(n) = \Omega(g_2(n)), g_2(n) = \Omega(g_3(n))$, etc. Your justification is not required to be formal/rigorous, e.g., you may state sentences like "since base of log does not matter in asymptotic notations…", etc.

Problem 3 (6 + 25 = 31 points)

[This problem is based on CLRS problem 2-4] Let $A[1:n]$ be an array of n distinct numbers. If $i < j$ and $A[i] > A[j]$, then the pair (i, j) is called an inversion of A.

(a) List the five inversions of the array $\langle 2, 3, 8, 6, 1 \rangle$.

(b) Give an algorithm that determines the number of inversions in an array in $O(n \lg n)$ time. (Hint: modify merge sort) Using recursion trees, justify the running time of your algorithm.

Problem 4 (15 points)

The following algorithm computes the n^{th} power of a number. More precisely, given a non-negative integer n and a number a, the algorithm returns a^n.

Algorithm 1 FAST-EXPONENTIATION(a, n)

Input: A number a and a non-negative integer n
Output: The value of a^n

```
1: if n = 0 then
2:     return 1
3: if n = 1 then
4:     return a
5: x = FAST-EXPONENTIATION(a, ⌊n/2⌋)
6: if n is even then
7:     return x · x
8: else
9:     return a · x · x
```

Prove the correctness of FAST-EXPONENTIATION procedure, i.e., show that FAST-EXPONENTIATION correctly returns the value of a^n. (Hint: use strong/complete induction, and example is given in: https://uncch.instructure.com/courses/34716/files/folder/Notes?preview=5058899)