Homework 1

CS6033 Design and Analysis of Algorithms I
Fall 2023
(Sec. B, Prof. Yi-Jen Chiang)

Due: Wed. 9/27 by 1pm
(submit online on NYU Brightspace; one submission per group)
Maximum Score: 100 points

Note: This assignment has 2 pages.

1. (25 points)
In the following, let f and g be positive increasing functions. Answer each question and **briefly justify your answers.** You get no points if you do not give a justification.

(a) Given that $f(n) = O(g(n))$, is it possible that $f(n) = \Omega(g(n))$? Is it always true that $f(n) = \Omega(g(n))$? **(5 points)**

(b) Given that $f(n) = o(g(n))$, is it possible that $f(n) = O(g(n))$? Is it always true that $f(n) = O(g(n))$? **(5 points)**

(c) Given that $f(n) = \Theta(g(n))$, is it possible that $f(n) = O(g(n))$? Is it always true that $f(n) = O(g(n))$? **(5 points)**

(d) Is it possible that $f(n) = \omega(g(n))$ and also $f(n) = O(g(n))$? **(5 points)**

(e) Is it possible that $f(n) + g(n) = \Theta(\min(f(n), g(n)))$? Is it always true that $f(n) + g(n) = \Theta(\min(f(n), g(n)))$? **(5 points)**

2. (6 points)
Give two **asymptotically different** functions, each of which belongs to both $\omega(n^3)$ and $o(n^4)$. Briefly justify your answers.

3. (20 points)
Problem 3-2 of the Textbook (p.71), **(c) - (f)**. Note that "A is O of B" means $A = O(B)$, and similarly for other notations ($o, \Omega, \omega, \Theta$). In your write up, copy and fill up the table with "Yes" or "No" in each table entry; after the table, **briefly justify your answer** for each sub-question. You get no points if you do not give a justification.
(**Note: 5 points for each of (c) - (f).**)

4. (24 points)
Let $S_1(n) = \sum_{k=1}^{n} k$, $S_2(n) = \sum_{k=1}^{n} k^2$ and $S_3(n) = \sum_{k=1}^{n} k^3$. In class, we already showed how to calculate $S_1(n)$ and $S_2(n)$. Your task here is to apply similar methods for $S_3(n)$.

(a) Without calculating the closed form, use rough estimations to derive a lower bound and an upper bound for $S_3(n)$, so that you can use them to express $S_3(n)$ in the $\Theta()$ notation. Give this $\Theta()$ notation in the **simplest form** (e.g., $\Theta(n)$ is in the simplest form but $\Theta(2n)$ is not). **(6 points)**

(b) Use the *perturbation method* (as discussed in class) to derive the closed form for $S_3(n)$.
(**Background Information:** You would need the formula of $S_1(n)$ and $S_2(n)$, which we already know: $S_1(n) = n(n+1)/2$ and $S_2(n) = n(n+1)(2n+1)/6$.) **(18 points)**

5. (25 points)
Given a sequence of n unordered real numbers, design and analyze an algorithm that finds **both** the **largest** and the **second largest** numbers among them using at most $3\lfloor n/2 \rfloor$ comparisons. Your algorithm should describe what to do when n is even and when n is odd. You need to show that your algorithm gives the correct answers, and that it uses at most $3\lfloor n/2 \rfloor$ comparisons.
(**Note:** Here *comparisons* refer to comparisons between any two items in the input sequence of n real numbers.)