Problem Set 4

This problem set is due **at 11:59pm** on **Tuesday, October 8, 2019.** (You have extra time for this problem set!)

Reading Assignment

Chapters 9 and 10 (skip 10.3)

Problem 4-1. Fibonacci Patterns [15 points]

The Fibonacci numbers $F_0, F_1, F_2, \ldots$, beginning with $0, 1, 1, 2, 3, 5, 8, 13, \ldots$, are defined recursively as follows:

$$F_n := \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{if } n = 1, \\ F_{n-1} + F_{n-2} & \text{if } n > 1. \end{cases}$$

These numbers satisfy many unexpected identities, such as $\sum_{i=0}^{n} F_i = F_{n+2} - 1$. (You don't need to prove this formula.)

(a) Guess a similar closed-form expression for

$$\sum_{i=0}^{n} F_i^2,$$

and prove your formula by induction. (Like the example, your closed-form expression may involve other Fibonacci numbers, but must avoid iterative processes like $\sum_{i=0}^{n}$ or $\prod_{i=1}^{n}$.)

(b) Use the perturbation method to compute the following sum, showing your work:

$$\sum_{i=0}^{\infty} \frac{F_i}{2^i}.$$

You may assume that the sum converges.

Problem 4-2. Asymptotic Notation [15 points]

(a) Indicate whether each function in the left column belongs to each set in the top row, by writing "YES" or "NO" in each cell. You do not need to prove your answers.

In the last row of the table, the function $a(n)$ is defined as follows: $a(1) = 2$, and $a(n) = (a(n-1))^{a(n-1)}$ for each integer $n \geq 2$.

	$\Theta(x^2)$	$O(2^x)$	$\omega(\log_2 x)$	$\Omega(e^{\sqrt{x}})$	$o(x)$
$\ln(x)$					
$\frac{1}{2}(x-1)^2$					
9					
2^{2x}					
$\sqrt{5x+3}$					
$a(x)$					

(b) Recall that $f = O(g)$ is defined as follows:

$$\exists c > 0.\, \exists n_0 \in \mathbb{N}.\, \forall n > n_0.\, |f(n)| \leq c \cdot g(n).$$

Use this definition to carefully prove that if $f(n) = O(g(n))$, then $(f(n))^2 = O((g(n))^2)$.

Problem 4-3. Integral method [10 points]

Use the integral method to give (and prove!) an estimate of the infinite sum

$$S = \sum_{i=1}^{\infty} \frac{1}{(2i-1)^3}$$

that differs from the true value of S by at most $\frac{1}{100}$.

Hint: Remove a few terms of S before approximating.

Problem 4-4. TriMergeSort [10 points]

Let's consider a new version of `MergeSort` called `TriMergeSort`, where the size n list is now broken into *three* sublists of size $n/3$, which are sorted recursively and then merged. Hopefully this new version will be faster! Since we know that floors and ceilings do not affect the asymptotic solution to a recurrence, let's assume that n is a power of 3.

(a) Show that $2n - 3$ comparisons are enough to merge three sorted lists each containing $n/3$ items. (For simplicity, we'll assume `TriMergeSort` uses *exactly* $2n-3$ comparisons to accomplish the merge.)

(b) If $T(n)$ is the number of comparisons that `TriMergeSort` uses to sort n items, it follows that $T(n) = 3T(n/3) + 2n - 3$, with $T(1) = 0$. Use the Plug and Chug method to find an *exact*, closed-form expression for $T(n)$ when n is a power of 3, and prove it by induction. Is this an asymptotic improvement over `MergeSort` (i.e., by more than constant factors)?