WEEK 02

1. Preparation for Assignment

If, and *only if* you can truthfully assert the truthfulness of each statement below are you ready to start the assignment.

1.1. **Reading Comprehension Self-Check.**

- I know that an algorithm's time efficiency is principally measured as a function of its input size by counting the number of times its basic operation is executed.
- I understand why it is **false** to say that a **basic operation** of an algorithm is the operation that contributes **least** toward the algorithm's running time.
- I understand that the established framework for analyzing an algorithm's time efficiency is primarily grounded in the order of growth of the algorithm's running time as its input size goes to infinity.
- I have pondered the idea that the main tools for analyzing the time efficiency of a non-recursive algorithm are to set up a sum expressing the number of executions of its basic operations and then to ascertain the sum's order of growth.
- I understand that there are several algorithms for computing the Fibonacci numbers with drastically different efficiencies.
- I will ponder the idea that the main tools for analyzing the time efficiency of a recursive algorithm are to set up a recurrence relation expressing the number of executions of its basic operations and then to ascertain the solution's order of growth.
- I know that *empirical analysis* is applicable to any algorithm.
- I know why it is *false* to say that *Algorithm Visualization*, or the use of images to convey useful information about algorithms, has three principal variations.
- I discovered a web site devoted to algorithm visualization/animation.

1.2. **Memory Self-Check.**

1.2.1. *Statements Regarding Algorithms.* Rank each quote in the About Algorithms reading. Rank them by your preference for them. Explain why your top and bottom ranked quotes were ranked as first and last.

Date: October 18, 2018.

1.2.2. *Pondering Brings Depth.* Ponder on the methodology you used rank the quotes. Convert the methodology into an algorithm. What is the $\mathcal{O}$ order of growth for your algorithm?

1.2.3. *Fill in the Blank.*

$$\lim_{n\to\infty}\frac{t(n)}{g(n)}=\begin{cases}0 & \text{the order of growth of t(n) \_\_\_\_\_ the order of growth of g(n)}\\ c>0 & \text{the order of growth of t(n) \_\_\_\_\_ the order of growth of g(n)}\\ \infty & \text{the order of growth of t(n) \_\_\_\_\_ the order of growth of g(n)}\end{cases}$$

1.2.4. *Apply Limits to Specific Function Pairs.* Calculate $\lim_{n\to\infty}\frac{t(n)}{g(n)}$ for each pair of functions.

$t(n)$	$g(n)$	$\lim_{n\to\infty}\frac{t(n)}{g(n)}$
$10n$	$2n^2$	
$n(n+1)/2$	n^2	
$\log_b n$	$\log_c n$	

Hint: L'Hôpital's rule says:

$$\lim_{n\to\infty}\frac{t(n)}{g(n)}=\lim_{n\to\infty}\frac{t'(n)}{g'(n)}$$

2. Week 2 Exercises

2.1. **Exercise 1 on page 50.**

Here is an example of a good answer for algorithm a:

Algorithm	**i**	
a	n	addit
b	the magnitude of n	multipl
c	n	compa
d	the sum of the magnitudes of two input numbers	r
e	the magnitude of n	elimination of a number from

2.2. **Exercise 8 on page 51.**

Here is an example of a good answer for function a:

$$\log_2 n.\ \text{Value change:}\log_2 4n-\log_2 n=(\log_2 4+\log_2 n)-\log_2 n=2$$

$$b)\sqrt{n}.\text{Value change:}\frac{\sqrt{4n}}{\sqrt{n}}n=2$$

$$c) n \text{Value change:} \frac{4n}{n} = 4$$

$$d) n^2.Valuechange : \frac{(4n)^2}{n^2} = 4^2$$

$$e) n^3.Valuechange : \frac{(4n)^3}{n^3} = 4^3$$

$$f) 2^n.Valuechange : \frac{2^{4n}}{2^n} = 2^{3n} = (2^n)^3$$

2.3. **Exercise 3 on page 59.**

Here is an example of a good answer for part a:
$(n^2+1)^{10}$. To prove this is in $\Theta(n^{20})$:

$$\lim_{n\to\infty} \frac{(n^2+1)^{10}}{n^{20}} = \lim_{n\to\infty} \frac{(n^2+1)^{10}}{(n^2)^{10}} = \lim_{n\to\infty} \left(\frac{n^2+1}{n^2}\right)^{10} = \lim_{n\to\infty} \left(1+\frac{1}{n^2}\right)^{10} = 1.$$

Or, informally: $(n^2+1)^{10} \approx (n^2)^{10} = n^{20} \in \Theta(n^{20})$

2.4. **Exercise 1 on page 67.**

Here are two examples of good answers for summation a:

$$1+3+5+7+\ldots+999 = \sum_{i=1}^{500}(2i-1) = \sum_{i=1}^{500} 2i - \sum_{i=1}^{500} 1 = 2\frac{500*501}{2} - 500 = 250,000.$$

Or, by using the formula for the sum of the arithmetic series with $a_1 = 1, a_n = 999, n = 500$,

$$\frac{(a_1+a_n)n}{2} = \frac{(1+999)500}{2} = 250,000.$$

2.5. **Exercise 2 on page 67.**

Here is an example of a good answer for part a:

$$\sum_{i=0}^{n-1}(i^2+1)^2 = \sum_{i=0}^{n-1}(i^4+2i^2+1) = \sum_{i=0}^{n-1} i^4 + 2\sum_{i=0}^{n-1} i^2 + \sum_{i=0}^{n-1} 1 \in \Theta(n^5)+\Theta(n^3)+\Theta(n) = \Theta(n^5)$$

$$(\text{or just} \sum_{i=0}^{n-1}(i^2+i)^2 \approx \sum_{i=0}^{n-1} i^4 \in \Theta(n^5)).$$

2.6. **Exercise 3 on page 76.**

3. Week 2 Problems

3.1. **Exercise 12 on page 61.**

```
The key idea here is to walk intermittently right and left
going each time exponentially farther from the initial position.
 A simple implementation of this idea is to do the following
until the door is reached:
```

3.2. **Analysis Using $\mathcal{O}(f(n))$.**

If it wasn't already part of your comparison/contrast of the three algorithms you created for the "Breaking Up" problem of last week's assignment, do a $\mathcal{O}(f(n))$ analysis of each algorithm here.

3.3. **Exercise 6 on page 83.**

3.4. **Exercise 9 on page 84.**

Hint: Use mathematical induction.