Chapter 01, 1.6 – Comparing Algorithms, Review Problem(s): ***41, 42, 43***

https://jcsites.juniata.edu/faculty/kruse/cs240/LectureNotes/algcomplex.htm

We will spend much more time on this in CS 315 – Algorithms and Analysis.

1. Present magazine or paperback.
2. Give the number of pages.
3. Ask for a page number from the class.
4. Sequential search.
5. Is there a better way?
6. Binary search demo.
7. Discuss changes in problem size.

Open a new Java class.

Write a simple for loop. Code a counter for the number of times the for loop executes, and print out the counter.

```
int counter = 0;
for (int i=0; i<100; i++)
{
	counter++;
}
System.out.println("The loop(s) executed "+counter+" times.");
```

Change the upper limit on the loop. How does that change the counter?

Now, add an inner loop, using the same upper limit, and keep the counter in the innermost loop. How does that change the counter?

```
	public static void main (String[] args)
	{
		int counter = 0;
		for (int i=0; i<100; i++)
		{
			for (int j=0; j<100; j++)
			{
				counter++;
			}
		}
		System.out.println("The loop(s) executed "+counter+" times.");
	}
```

Try some big numbers for the upper limit, keeping it the same on each loop.

For each, the nested loops and the single loop, can you determine a mathematical function for the counter, as a function of ***N***, the upper limit?

Finally, add a new loop, counting the number of times it executes, runs logarithmically (think binary search and divide by 2), rather than linearly.

```
counter = 0;
int value = 100000;
while (value>1)
{
	value = value / 2;
	counter++;
}
System.out.println("The loop(s) executed "+counter+" times.");
```

We characterize the execution of algorithms mathematically, as functions of the problem size ***N***, which here was the upper limit on the loop(s).

- Why did the single loop execute on the order of ***N***?
- Why did the nested loop execute on the order of N^2 ?
- And Why did the last example execute on the of $\boldsymbol{log\ N}$?

We use *O notation* for this, where given the mathematical function of ***N***, we disregard any lower order terms, and any coefficients on the highest order terms. The examples here were simple and didn't have any lower order terms or coefficients, but several problems in the textbook provide practice on this.

So, here we had examples of code which were running at $O(N), O(N^2), O(\log N)$.