Notes - Week 1 - August 25, 2021

- An **algorithm** is a finite step by step process to solve a problem
  - **finite** it must terminate given **any** input
- What is the Analysis of Algorithms?
  - How long does it take to run?
  - How much space does it use?
- Most research is on the time it takes an algorithm to run. Memory is "cheap".
- Typically concerned with large data sets.
  - Most algorithms will seem fast in small data sets
  - Large data sets simulate real situations
- We always analyze an algorithm before we write it. **Analysis**
- Trying it out is referred to as **Measuring**.
- **Big O Notation** refers to the time complexity of a statement, notated as $O(x)$ where x is the amount of statements the algorithm executes, based on the variable n.
  - $O(n^2)$
  - $O(log(n))$
  - $O(1)$
  - etc

The Selection Problem

- We have n numbers, in an array, no guarantee of sorting.
- We want the k^{th} largest number.
- Algorithm (0): $O(n^2)$ complexity
- Algorithm (1): $O(nlog(n))$ complexity
- New Algorithm:
  1. Read the first k numbers into a separate array
  2. Sort the new array
  3. Read the rest of the numbers. If they're smaller, replace the largest number in the array with an insertion.
- Is this new algorithm faster?
  - Some of the time
  - depends on how large k is relative to n
- Example: Find the 5^{th} largest number in an array of size $1,000,000$
  - $n = 1,000,000$
  - $k = 5$
  - Using $nlog(n)$ sort, it's approximately $20,000,000$ statements
  - Using new algorithm, it's about $5log(5) + 500,000 + 5(500,000) \ = 3,000,000$ statements
- As k approaches n, the new algorithm approaches the original time complexity of $nlog(n)$.