Algorithmic Complexity

**findItem(A,v) //A is the array & v is the value**

**index = -1**

**for i = 1 to A.length**

**if A[i] == v**

**index = i**

**return index**

Lots of simplifications in how this is done

We assume that each line was equal to a cost of one, however that is not really true each instruction may have its own cost. Also depends on the technology, language

We need a way to look at this that is language and technology independent

Asymptotic analysis: As A.length goes to infinity how does the algorithm perform.

Suppose we look at an inventory system. Assuming that it taskes 10,000 ms to read initial data from a file and an extra 10ms for each line in the file. Process n transactions in 10,000+10n. In computer science to do this we use Big-O notation: How did two functions grow as n goes to infinity?

Let n be the size of the programs input (nothing specific on the data type)

Let T(n) be a function that represents an algorithms precise runtime for some given input of size n.

Let f(n) be a simple growth function such that f(n)=n

We can express growth rate of T(n) by relating another function.

If T(n) grows no faster than f(n), the we say:

T(n) is in O(f(n)) //Order of f(n) or constant = O

Or T(n) is in O(n).

If and only if T(n) <= c.f(n), c is a constant

Whenever n is big, for some large C.

Inventory:  
T(n) = 10000+10n

F(n) = n

T(n) <= c.f(n), as n approaches infinity

n=10000

20n f(n).20

10000+10n = T(n)

T(n) will grow faster than c.f(n). large n, startup cost may be worth it small n, a less efficient may be faster than large startup costs.

Going from cost to Big-O notation

T(n) what is the term in that equation that is going to grow the fastest.

T(n) = 5n^3+3n^2+n+5

So here look at only the term n^3

Two steps to go from cost to order.

* If there are multiple terms keep the one with the largest growth rate.
* Discard the constants.

T(n) is in O(n^3), going to give us the upper bound on how T(n) grows as n approaches infinity.

Don’t need to know what c is just need to know that there is some c that works