# Logarithms

It is important to understand deep in your bones what logarithms are and where they come from.

A logarithm is simply an inverse exponential function. Saying $b^x = y$ is equivalent to saying that $x = \log_b y$.

Logarithms reflect how many times we can double something until we get to $n$, or halve something until we get to $1$.

# Binary Search

In binary search we throw away half the possible number of keys after each comparison. Thus twenty comparisons suffice to find any name in the million-name Manhattan phone book!

How many time can we halve $n$ before getting to $1$?

Answer: $\lceil \lg n \rceil$.

# Logarithms and Trees

How tall a binary tree do we need until we have $n$ leaves?
The number of potential leaves doubles with each level.
How many times can we double 1 until we get to $n$?
Answer: $\lceil \lg n \rceil$.

# Logarithms and Bits

How many bits do you need to represent the numbers from $0$ to $2^i - 1$?

Each bit you add doubles the possible number of bit patterns, so the number of bits equals $\lg(2^i) = i$.

# Logarithms and Multiplication

Recall that

$$\log_a(xy) = \log_a(x) + \log_a(y)$$

This is how people used to multiply before calculators, and remains useful for analysis.
What if $x = a$?

# The Base is not Asymptotically Important

Recall the definition, $c^{\log_c x} = x$ and that

$$\log_b a = \frac{\log_c a}{\log_c b}$$

Thus $\log_2 n = (1/\log_{100} 2) \times \log_{100} n$. Since $1/\log_{100} 2 = 6.643$ is just a constant, it does not matter in the Big Oh.