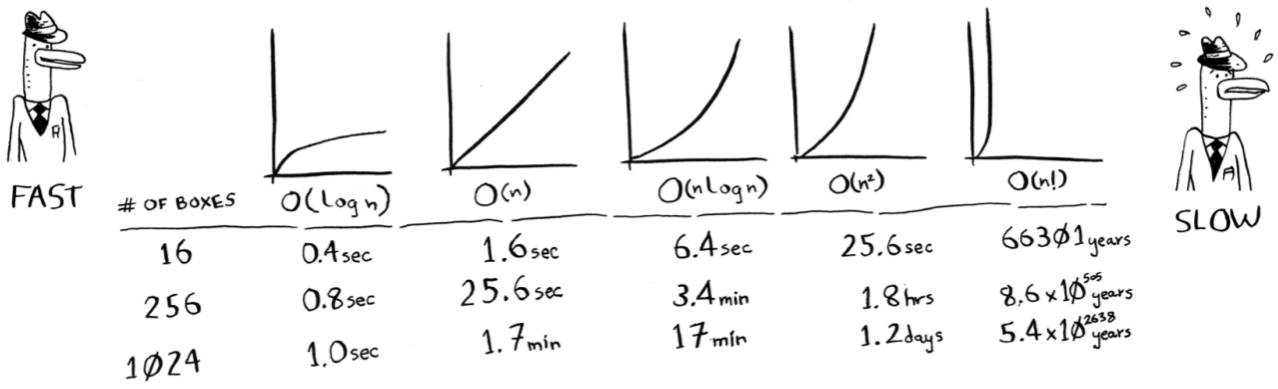
Big-o-notation

- Big-o-notation is special notation that tells you how fast an algorithm is.
- Simple search needs to check each element, so it will take **n** operations. The run tme in Big O notation is **O(n)**.
- Binary search needs **log(n)** operations to check a list of size n. Running time in Big O notation is **O(log n)**.
- Big O will not tell you anything about how long it will take, it will only tell you how fast the algorithm grows.
- Big O always considers the worst case. If we are lucky, we can find the value that we are searching for even in the first try, but Big(O) always considers the worst case. Omega(n) considers the best case, and Theta(n) considers the average case.



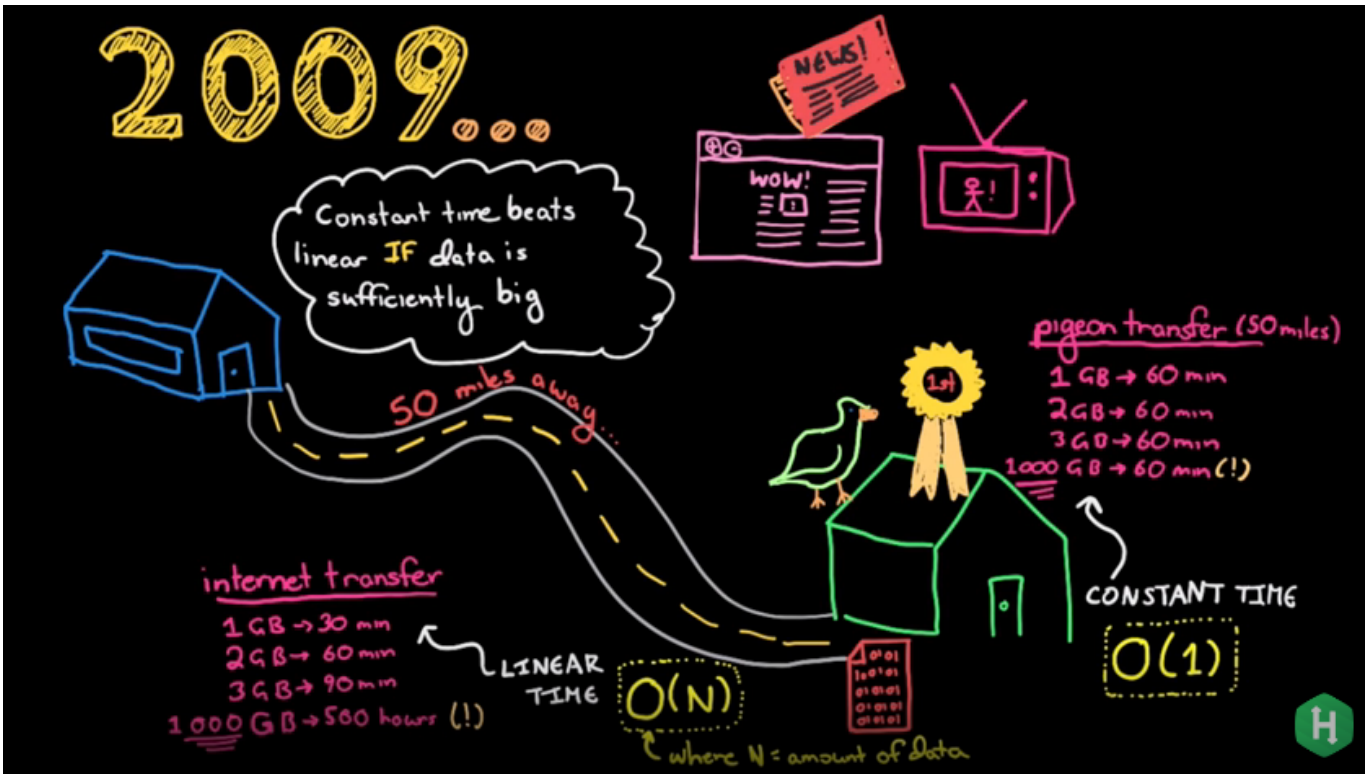
Some Common Big O Run Times

- O($\log_n$), also known as log time. Binary search.
- O(n), also known as linear time. Simple search.
- O(n * $\log_n$). A fast sorting algorithm like quicksort.
- O(n^2). A slow sorting algorithm like selection sort.
- 0(n!). A really slow algorithm like traveling salesperson.

Summary

- Algorithm speed isn't measured in seconds, but in growth of the number of operations.
- It says how time scales with respect to some variables.
- We talk about how quickly the run time of an algorithm increases as the size of the input increases.
- Run time of algorithms is expressed in Big O notation.

Pigeon and Internet Example



Algorithm Analysis

Example 1

```
for(int i = 0; i<n; i++){
    m = m + 2;
}
```

Loop happens ***n*** times

Constant Operation

The leading constant is ignored!

$f(n) = c \times n$

constant

loop

$f(n) = O(n)$

Algorithm Analysis Part 1

Example 3 – Nested Loops

```
for(int i = 0; i<n; i++){
    for(int j = 0; j<n; j++){
        k=k+1;
    }
}
```

Outside loop has inside loop happen ***n*** times!

Inside Loop goes to ***n***

Constant time operation

$f(n) = c \times n \times n$

Constant

Inner Loop

Outer Loop

computer science BREAKDOWN

$f(n) = c \times n^2 \longrightarrow f(n) = O(n^2)$

Example 4 – Consecutive Statements

```
for(int l = 0; l<n; l++){
    m = m + 5;
}
for(int i = 0; i<n; i++){
    for(int j = 0; j<n; j++){
        k=k+1;
    }
}
```

1

2

Single Loop Operation

Nested Loops

$f(n) = c \times n + c \times n \times n$

1 2

$f(n) = c \times n + c \times n^2$

computer science BREAKDOWN

$f(n) = O(n^2)$

Algorithm Analysis Part 1

Example 5 – If-Then-Else

```
if( x + 1 < 5 ){
    return -1;
} else {
    for(int i = 0; i<n; i++){
        for(int j = 0; j<n; j++){
           k=k+1;
        }
    }
    return k;
}
```

Constant

Constant

Outer Loop

Inner Loop

Constant

$f(n) = c_0 + c_1 + n \times n \times c_2$

$f(n) = c_0 + c_1 + c_2 \times n^2$

$f(n) = O(n^2)$