Hashing Strings

- option 1
  - sum the characters

e.g.

"dog" → 100 + 111 + 103 = 314

"pig" → 112 + 105 + 103 = 320

"god" → 103 + 111 + 100 = 314

- option 2
  - multiply by $31^{(length-pos-1)}$

"dog" → $100 \times 31^2 + 111 \times 31 + 103$

$= 99,644$

"pig" → $112 \times 31^2 + 105 \times 31 + 103$

$= 110,990$

"god" → $103 \times 31^2 + 111 \times 31 + 100$

$= 102,524$

- Hashing Objects
  - be sure you hash the part of the object that identifies it and won't change
  - once you identify these fields, combine them
  - if small, use the approach for string
  - if large you can XOR


```
  101100
XOR 011110
  110010
```

- operations
  - simple statements

1 operation each line

```
int x = 4;
y = x * x + 7;
z = (x - y) / 42 + 16;
```

  - decisions
    - consider best & worst case

```
if ( ---- )
```

usually worst case →) ~ 7n+4 operations

```
else
```

best case →) ~ 12 operations

- loops
  - find # of iterations
  - count operations inside

n iterations

```
for(int i = 0; i < n; i++)
{
    ) 7 operations
}
```

  - total is operations × iterations

e.g. $n \times 7 = 7n$

- function calls
  - our function → analyze it
  - system function → assume 1 operation

- operations → Big-Oh

$f(n) = n^2 + 2n + 451$ $f(n) = O(?)$

$f(n) = O(n^2) + O(n) + O(1)$

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

$f(n) = n^3 + 1$

$f(n) = O(n^3) + O(1)$

$f(n) = O(n^3)$

$\log_2 x = y$ $\log_{10} x = y$

$2^y = x$ $10^y = x$

$\log_2 72 = x$

$2^x = 72$

$\log_{10} 2^x = \log_{10} 72$

$x \log_{10} 2 = \log_{10} 72$

$x = \frac{\log_{10} 72}{\log_{10} 2}$

$\log_a b = \frac{\lg b}{\lg a}$

$\log_a b = \left(\frac{1}{\lg a}\right) \lg b$

↑ constant