Asymptotics:

ONE eqn to always remember:

$$\text{Runtime} = \text{Sum over \# layers}\left[\frac{\#\text{nodes}}{\text{layers}} \cdot \left(\frac{\text{amount of work}}{1 \text{ node}}\right)\right]$$

Also, remember these formulas:

$$\sum_{i=1}^{N} i = 1+2+3+\ldots+N = \frac{N(N+1)}{2} = \frac{N^2+N}{2}$$

$$\sum_{i=0}^{N-1} 2^i = 1+2+4+\ldots+2^{N-1} = 2\cdot 2^{N-1} - 1 = 2^N - 1$$

A few orders of growth in increasing order:

$\log(N)$, N, $N\log(N)$, N^2, 2^N, $N!$

Remember:

- Drop constants.
- Only keep the largest one!

Asymptotic Notation:

- Ω = Big Omega = describes the lower bound of a function
- O = Big O = describes upper bound of a function.
- Θ = Big Θ = describes tight bound of a function (upper & lower bounds are same)

Examples:

$n^3 \in \Omega(n^2)$ because runtime always $< n^3$

$n^2 \in O(n^3)$ because runtime always $> n^2$

$n^2 \in \Theta(n^2)$ because runtime always runs in n^2 time.

✡ Remember O & Ω are not best and worst case, as they are not super informative!

Tilde notation:

Same thing as orders of growth, but just keep the constant.

So, $15n^2 + 100n + 100 \sim 15n^2$

To check if 2 functions are the same runtime,

$$f(x) \sim g(x) \text{ if } \lim_{x\to\infty} \frac{f(x)}{g(x)} = 1$$

Amortized Runtime

- If you do a burst of work, you can "ignore" it.

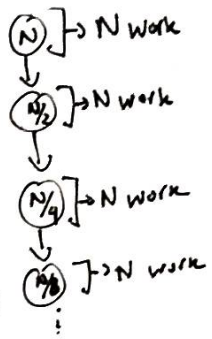
$$\frac{(1+1+1+\dots)+N}{N} \to \frac{N+N}{N} \to \frac{2N}{N} \to 2 \to \Theta(1)$$

Basic Runtime Calculations:

* Remember those two equations @ the front.

Complicated Runtimes:

```
public void stepper(int N){
    int q = 0;
    for (int i = 0; i<N; i++){
        q++;
    }
    stepper(N/2);
}
```



log(N) height

$$\sum_{i=0}^{\log N} \frac{N}{2^i} * 1 = N + \frac{N}{2} + \dots \frac{N}{2^{\log N}}$$

$$\Theta(2N) = \boxed{\Theta(N)}$$

```
public void (int N){
    int q = 0;
    for (int i=0; i<N; i++){
        for(int j=0; j<i; j++){
            q++;
        }}}
```

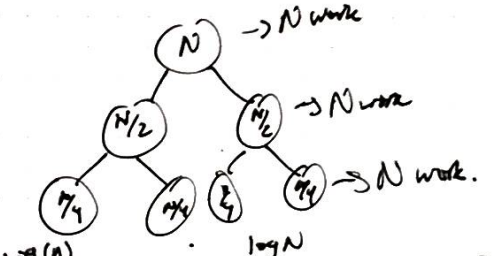
N times

$$1+2+3+4+\dots+N = \boxed{N^2}$$

Strategies

For Recursion, draw a tree!

```
public void grewt(int N){
    for (int i = 0; i<N; i++){
        System.out.println("I am growing");
    }
    grewt(N/2);
    grewt(N/2);
}
```



$$\sum_{i=0}^{\log(N)} \frac{N}{2^i} 2^i \to \sum_{0}^{\log N} N \to \Theta(N \log N)$$