0. rewrite for any sub. for *n*
1. rewrite with one back sub. then sub. that into original
2. rewrite for 2 back sub. then sub. that into original from 1
3. generalize to *i*
4. undo sub. for *n* if needed then turn remaining recursive call into the base case (e.g. set n-i=0)
5. sub. in base case + solve

largest size n of a problem that can be solved in 1 minute with running time n²

- n^2 ≤ 60000 ms = 1 min
- n ≤ √60000 = 244.94,
- always round down to 244 = n

3n-4 ∈ Ω(n²); prove false by contradiction

- assume that cn² ≤ 3n-4 ≤ 3n for all n ≥ n_0
- cn² ≤ 3n
- cn ≤ 3
- n ≤ 3/c for all n ≥ n_0; impossible

$$\mathbf{x(n)= 2x(n/2) + n,\ \ x(1) = 1}$$

Step 1 - replace n by n/2

$$x(n/2) = \boxed{2x(n/4) + (n/2)}$$

$$\begin{aligned} x(n) &= 2x(n/2) + n \\ &= 2[2x(n/4) + (n/2)] + n \\ &= 4x(n/4) + 2(n/2) + n \\ &= 4x(n/4) + 2n \end{aligned}$$

Step 2 - replace n by n/4

$$x(n/4) = \boxed{2x(n/8) + (n/4)}$$

$$\begin{aligned} x(n) &= 4x(n/4) + 2n \\ &= 4[2x(n/8) + (n/4)] + 2n \\ &= 8x(n/8) + 4(n/4) + 2n \\ &= 8x(n/8) + 3n \end{aligned}$$

p 3 - $x(n) = 2^k x(\frac{n}{2^k}) + kn$

4 - want $x(1)$, so let $n = 2^k \rightarrow \mathbf{k = lg(n)}$

p 5 - $x(n) = 2^{lg(n)} \cdot x(\frac{2^k}{2^k}) + lg(n) \cdot n$

$$\begin{aligned} &= n \cdot x(1) + n \cdot lg(n) \\ &= \mathbf{nlg(n) + n} \end{aligned}$$

$\mathbf{x(n)= x(\frac{n}{2}) + 1,\ x(1) = 1}$ <u>Binary Search</u>

Solve for $n = 2^k$

Step 0 - Rewrite the recurrence, making the su

$$x(2^k) = x(2^{k-1}) + 1$$

Step 1 - replace 2^k with 2^{k-1}

$$\begin{aligned} x(2^{k-1}) &= x(2^{k-2}) + 1 \\ x(2^k) &= x(2^{k-2}) + 1 + 1 \\ &= x(2^{k-2}) + 2 \end{aligned}$$

Step 2 - replace 2^k with 2^{k-2}

$$\begin{aligned} x(2^{k-2}) &= x(2^{k-3}) + 1 \\ x(2^k) &= x(2^{k-3}) + 1 + 2 \\ &= x(2^{k-3}) + 3 \end{aligned}$$

3 - $x(2^k) = x(2^{k-i}) + i$

p 4 - $2^{k-i} = 1$

$$\begin{aligned} 2^{k-i} &= 2^0 \\ k - i &= 0 \\ i &= k \end{aligned}$$

5 - $x(2^k) = x(2^{k-i}) + i$

$$\begin{aligned} &= x(2^{k-k}) + k \\ &= x(2^0) + k \\ &= x(1) + k \\ &= 1 + k \end{aligned}$$

$$\begin{aligned} n &= 2^k \\ k &= lg(n) \end{aligned}$$

$$\begin{aligned} x(n) &= 1 + lg(n) \\ &= \mathbf{lg(n) + 1} \end{aligned}$$

create and clean up a student object on the stack then the heap

```
Student a; // local stack var.
Student *b = new Student(); //global heap var.
delete b;
```

selection sort scans for the minimum element and knows where to put it
insertion sort knows the next-smallest element and scans for its place
bubble sort
ANY MIGHT BE SYMMETRIC

$\log_2 n^{\log_2 17}$ is $\Theta(\log_2 17^{\log_2 n})$

1. constant
2. 3rd root < sqrt(x)
3. log
4. linear n
5. n^k < n^k log n < n^(k+1) log n
6. exponential k^n
7. factorial n!
8. n^n

Is $3n - 4 \in \Omega(n^2)$? Circle your answer: yes / $\boxed{\textit{no}}$. (2 points)

If $3n - 4 \in \Omega(n^2)$, then $\exists c \in \mathbb{Z}$ where $0 \leq cn^2 \leq 3n - 4$ $(\forall n \geq n_0)$.
We also have $3n - 4 \leq 3n + 4n = 7n$
Combining these statements gives us:

$$cn^2 \leq 7n$$
$$cn \leq 7$$
$$n \leq \frac{7}{c}$$

Because n can grow to infinity, there is no way to bound it by any constant $\frac{7}{c}$. $\therefore 3n - 4 \notin \Omega(n^2)$.