When considering growth rates, we...

- Focus on dominant term, and ignore constant factors.
- We do NOT ignore non-constant multipliers.

Example Big-Oh proof

$$3n+7 = O(n)$$

$$\forall n > n_0 \ \exists c : 3n+7 \le c \cdot n$$

$$\equiv \frac{3n+7}{n} \le c$$

n	$f(n) = 3n+7$	$g(n) = n$	$\frac{f(n)}{g(n)}$
1	10	1	10
10	37	10	3.7
100	307	100	3.07 = c

- If the equality is true, you will be able to find some c such that the ratio will always be less than c.
- If you cannot find a c that is greater than all the ratios then the equality does not hold.
- Ideally pick a c as close to the top of the table as you can.

hypothesis

$3n+7$

$\boxed{n \ge 1}$ $\quad 3n+7 \overset{???}{\le} 10n$

$1 \le n$

$7 \le 7n$

$3n+7 \le 3n+7n = 10n$

$\boxed{\text{Ex}}$ big Ω proof

Use theorem: $f = O(g)$

$\equiv g = \Omega(f)$

$(n+1)^3 = \Omega(n^2)$

$\equiv n^2 = O((n+1)^3)$

$\equiv n^2 = O(n^3)$

$n^2 \leq c \cdot n^3$

n	$f(n)$ n^2	$g(n)$ n^3	$\frac{f(n)}{g(n)}$
1	1	1	1
10	100	1000	$\frac{1}{10}$
100	...	...	$\frac{1}{100}$

$n_0 = 1$, $c = 1$

$n \geq 1 \rightarrow n^2 \leq n^3$

$1 \leq n$

$n \leq n^2$

$n^2 \leq n^3$

- Big Oh: "worst case scenario..."
- Big Omega: "best case scenario..."
- Big theta: "the time complexity is exactly"

Heuristics for analyzing time complexity of pseudocode

- **Basic Principles**
  - Count every "constant operation" as 1.
  - Anything irrelevant to input size counts as 1.

- We will get some function $f(n)$ (n is the size of the input).

ICE

procedure closest-pair($(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$: pairs of real numbers)

min = ∞

for $i = 2$ to n ---------- $(n-1)$ iterations

  for $j = 1$ to $(i-1)$ ---------- $(i-1)$ iterations

    if $(x_j - x_i)^2 + (y_j - y_i)^2 <$ min THEN

      min = $(x_j - x_i)^2 + (y_j - y_i)^2$ $\longrightarrow$ 1

      closest pair = $((x_i, y_i), (x_j, y_j))$

return closest pair

$$T(n) = \sum_{i=2}^{n} \left(\sum_{j=1}^{i-1} 1 \right) = \sum_{i=2}^{n} (i-1) = \frac{n(n-1)}{2} = \Theta(n^2)$$