big-O Notation

algorithm classification based on how the runtime of the algorithm changes w.r.t. the input size.

Let f ang g be two functions

$$f, g : \mathbb{N} \rightarrow \mathbb{R}^+$$

We say that

$$f(n) \in O(g(n))$$

if $\exists c \in \mathbb{R}^+, n_0 \in \mathbb{N}$ s.t.

$$\forall n > n_0 : f(n) < c g(n)$$

Ex) Let's prove $3n^2 + 2n + 3 \in O(n^2)$

We need to find c and n_0 s.t.

$$\forall n > n_0 : 3n^2 + 2n + 3 < c n^2$$

Let $c = 4$.

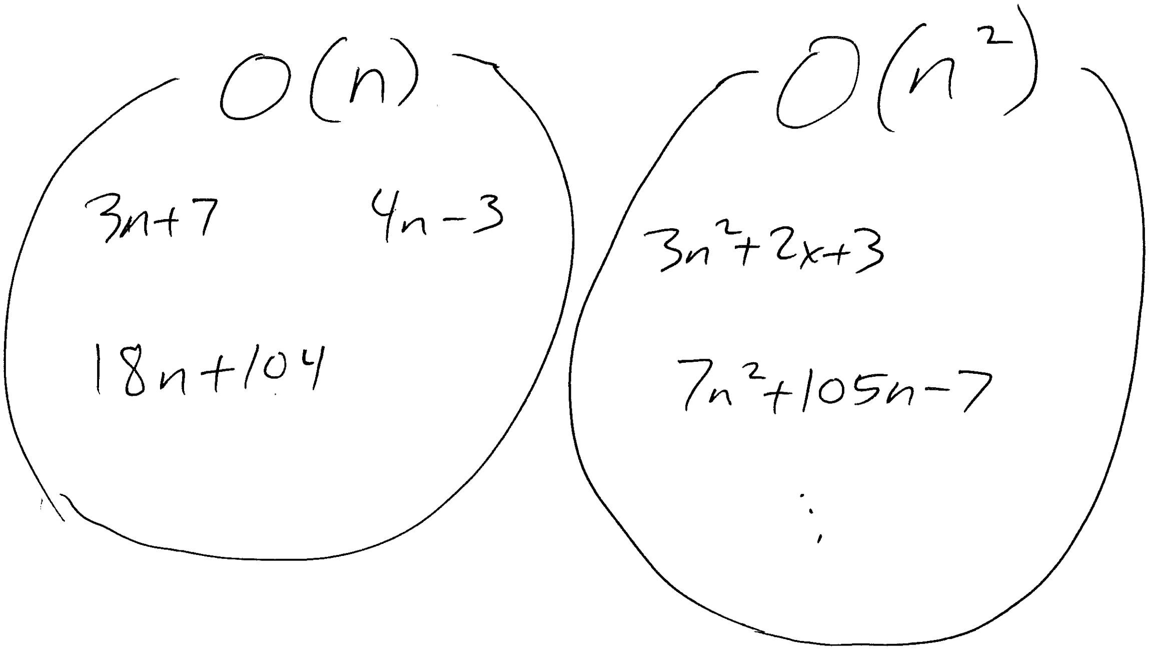
$$3n^2 + 2n + 3 < 4n^2$$

$$2n + 3 < n^2$$

$$\vdots$$

$$n_0 = 3$$

Let's cheat:

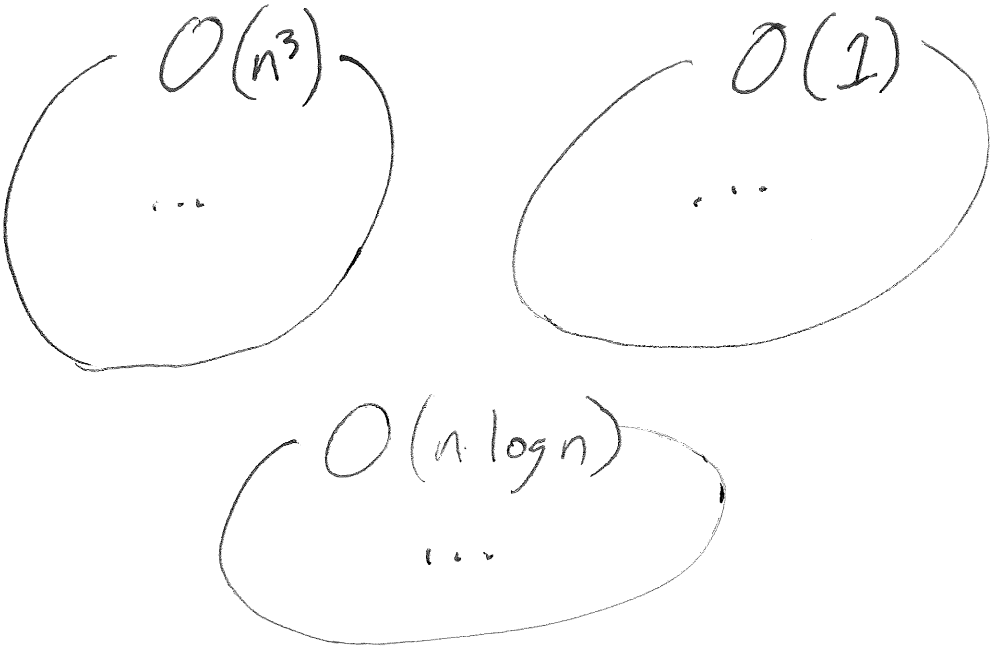


↑ draw those on the board.

Do you see the pattern?

- for polynomials, just keep the high-order term w/out the coefficient.

Have the students fill in these ↓


$O(n^3)$
...
$O(1)$
...
$O(n \cdot \log n)$
...