$\{x \in \mathbb{R} \mid x^3 \geq 1\} = \{x \in \mathbb{R} \mid x \geq 1\}$

$\{y \in \mathbb{R} \mid y^2 = 2\} = \{\sqrt{2}, -\sqrt{2}\}$

$\{y \in \mathbb{Z} \mid y^2 = 2\} = \emptyset = \{\}$

$\{z \in \mathbb{Z} \mid z < z^2\} = \{z \in \mathbb{Z} \mid z \neq 0, z \neq 1\}$

$= \mathbb{Z} - \{0, 1\}$

$A = \{a, b, c, d, e\}$

$B = \{a, b, c, d, e, f, g, h\} = A \cup \{f, g, h\}$

Q: $A - B = \{\}$

$\{\ \{1, 2, 3\}\ \}$

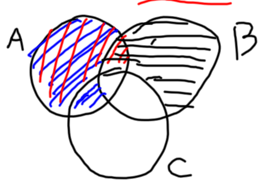
$\{\ \{\}\ \} \neq \{\}$

$A - B \neq B - A$ in general!

$A \times B \overset{?}{\neq} B \times A$

$A = \{vw, saab\}$
$B = \{ian\}$
$\langle vw, ian \rangle \in A \times B$

$(A - B) \cup (A - C) \cup (B - C)$



$U = \{vw, saab, bmw, audi\}$

$\{vw, bmw\} = 1010$

Show $(p \leftrightarrow q) \equiv (\neg p \leftrightarrow \neg q)$.

$a \leftrightarrow b$

$\equiv \boxed{(a \to b)} \wedge (b \to a)$ — T8, R1: defn $\leftrightarrow$

$\equiv \boxed{(\neg b \to \neg a)} \wedge (b \to a)$ — T7, R2 w $p = a$, $q = b$:

$\equiv \underbrace{(\neg b \to \neg a)}_{p} \wedge \underbrace{(\neg a \to \neg b)}_{q}$ — T7, R2 w $p = b$, $q = a$

$\equiv (\neg a \to \neg b) \wedge (\neg b \to \neg a)$ — T6 $\wedge$ commutes $p = \neg b \to \neg a$, $q = \neg a \to \neg b$

$\equiv \neg a \leftrightarrow \neg b$ — T8, R1 w/ p being $\neg a$, and q being $\neg b$,

TFAE: (The Following Are Equivalent)
- f is onto
- f is invertable.

a.k.a. $\forall x,\ g(f(x)) = x$

a.k.a. $g \circ f = I$ ← the identity function

Some functions:

$\lceil x \rceil$ – ceiling

$\lfloor x \rfloor$ – floor: $\lfloor k + \varepsilon \rfloor = k$, where $k \in \mathbb{Z}$, $\varepsilon \in [0,1)$

nod_{10} – number of digits

$nod_{10}(1000) = 3$

$nod_{10}(999) = 3$

$nod_{10}(500) = 2\frac{1}{2}$?

$\log_{10}$ → $nod_{10}(10^3) = 3$

$nod_{10}(10^3 \cdot 10^5) = 8$

$nod_{10}(10^x \cdot 10^y) = nod(10^x) + nod(10^y)$
$= x + y$

$\log(a \cdot b) = \log(a) + \log(b)$

Find whether a list l contains a number a:

$l = (a_0, a_1, a_2, a_3, \ldots, a_{n-1})$

for $i = 0$ to $n-1$ for each $\{i \mid i \geq 0, \wedge\, i \leq n-1\}$

  if $(a_i \stackrel{?}{=} a)$ return true   a.equals(a_i)

return false

How many steps?

n-1 steps?   1 step?   0 steps?

---

binary search: as above,

→ but $a_0, a_1, \ldots, a_{n-1}$ are in order (non-decreasing)

$5 \in \{-7, -2, 0, 3, 6, 10, 15, 21, 30\}$

How many steps in general?

To find whether $a \in \{a_{lo}, \ldots a_{hi-1}\}$

Check: is $a \leq a_{\frac{hi-lo}{2}}$?

If so: $hi = \frac{hi-lo}{2}$

else $lo = \frac{hi-lo}{2}$

Regardless: the active range of possible indices is cut in half each step.

We stop when $hi - lo = 1$.

How many times can we divide n by 2, before we reach 1?

Call this functions

numChecks(9) =

1+ numChecks(~~4~~ 5)

= 1+ (1+ numChecks(2))

= 1+ (1+ (1+ numChecks(1)))

= 1 (1 + (1+1)) = 4

---

numChecks(8) =

1+ numChecks(4)

= 1+3 =

n	numChecks(n)
8	3+1 = 4
16	5
32	6
64	7
100	8
128	8

1000 ...

10000

100000

1000000

$nod_2(8) = 3$

$nod_2(16) = 4$

$nod_2(32) = 5$

$$numChecks(n) = nod_2(n) + 1$$

# of steps for numChecks(n) $\cong \log_2(n) + 1$

26 +1

1100110[illegible]11101~~0~~

$2^{10} = 1024$

$2^{20} = 2^{10} \cdot 2^{10} \cong 1000 \cdot 1000 = 1{,}000{,}000$

$2^{20} \cdot 1000 \cong 2^{20} \cdot 2^{10} = 2^{30} \equiv$ Gig

10^{80} = # of particles

$\cong (2^{3.27})^{80} \cong 2^{250}$

Upshot: To find an object in an unsorted list takes about n steps (where n is the size of the list).

In a sorted list, it takes about $\log_2(n)$ steps. "$\lg(n)$"

Why "about"?

- I want to talk about the steps of algorithm, not the assembly language steps.
- being off by a constant factor is no worse than switching to a different assembly language, or a machine a few years old.
- For large inputs, startup overhead is insignificant

We say A function f is "big-Oh" of another function g if:

$\boxed{f = O(g)}$ means

$f \le g$
- up to constant factor
- for "big" inputs

$\exists N_0 > 0$

$\exists c > 0 . \left(\forall n \in \mathbb{N}. \; f(n) \le c \cdot g(n) \right)$

$n > N_0$

$numChecks(\cdot) = O(\log(n))$

setup overhead

Is this true? $numChecks(n) = 3 \cdot \log_2(n) + 1 + 7$

That is,

$$\boxed{3\log_2(n) + 8 \le c \cdot \log(n)}$$

Try: $c = 17$

$3\log_2(n) + 8 \le 17\log_2(n)$

$= 9\log_2(n) + 8\log_2(n)$

True if $n > 2$

Let $N_0 = 2$.

Let $f(n) = 5n^2 + 3$

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

To show: We'll find c, N_0 ∋. "such that" "s.t."

$\forall n > N_0 \quad 5n^2 + 3 \le c \cdot n^3$.

Try $\boxed{\begin{matrix} c = 6 \\ N_0 = 2 \end{matrix}}$

$5n^2 + 3 \le 6n^3 = 5n^3 + n^3$

True if $n^3 \ge 3$
$n \ge 2$

$n \ge 2$
$n^3 \ge 8$

$5n^2 + 3 \le 5n^3 + \underline{8} \le 5n^3 + \underline{n^3} = 6n^3$

f is O(g) means "f ≤ g" with those 2 caveats.