Proofs of Correctness

1. Termination
   - → OPT 1: give an upper bound, of the worst-case runtime
   - → $\boxed{\text{OPT 2}}$: prove it eventually returns. (who knows how long)
2. Partial Correctness
   - → recursion / loop invariants
   - → "if it terminates, then it is correct"

Let $(S, \leq)$ be a poset

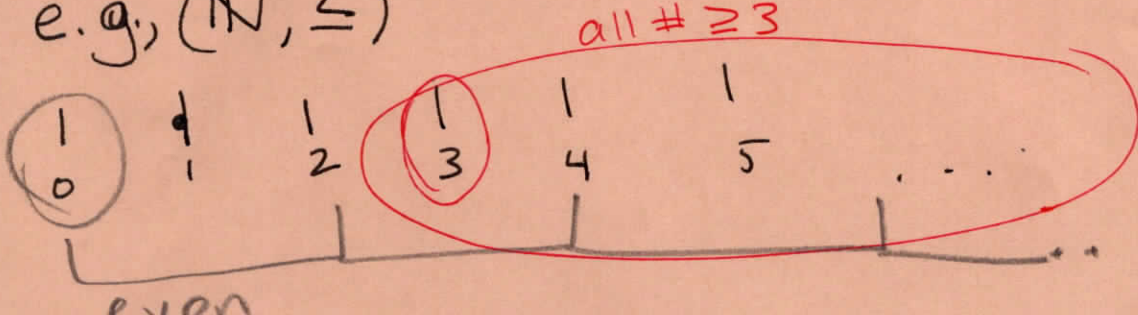
↳ e.g., $(\mathbb{N}, \leq)$ e.g., $(\mathbb{R}, \leq)$

e.g., (DAG, parent/child)

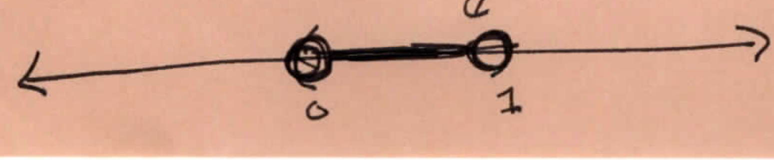
We say that S is <u>well-ordered</u> iff every subset $S' \subseteq S$ has a "least" element.

That is, $\forall\, S' \subseteq S$, $\underbrace{\exists!}_{\text{there exists a unique}}\ e \in S'$ s.t. $\forall\, s' \in S'$, $e \leq s'$.

e.g., $(\mathbb{N}, \leq)$



nonexample: $(\mathbb{R}, \leq)$ ↙ the open interval $(0,1)$ does not have a smallest #!



Note: $0 \notin (0,1)$

[0,1] has a smallest element, but is not a well-ordered set b/c

$S = [0,1]$

$S' = (0,1)$ ← does not have a smallest elt!

---

reminds me of the diff btwn inf & min

inf $(0,1) = 0$ ← can go outside the set.

min $(0,1)$ DNE

inf $[0,1] = 0$ must be in the set

min $[0,1] = 0$

TO prove that an algorithm terminates, define a decrementing fcn

$d : \mathcal{S} \longrightarrow S$ ← a well-ordered set

↑ the state space

such that

1. If there is a loop, d strictly decreases between iterations of the loop
2. If there is recursion, d must decrease as we make the recursive call.

If such a fcn exists, then our algorithm terminates! Why? Consider all values of d throughout the execution. call that set S'.

has a smallest val ⟹ no more recursion, looping!

Example:

```
int i = 1.
While i ≤ 10
| print i
| i++
end while
```

→ note: result of this is to print the #s 1 through 10.

Consider the decrementing fcn

$$d : S \longrightarrow \mathbb{N}$$

defined by $d(*) = 10 - i$ } common trick!

Since $\mathbb{N}$ is well-ordered, $\min(\text{im}(d))$ is realized, which means that the algorithm terminates. $\boxed{}$

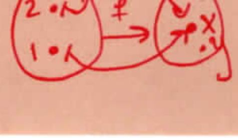
$$f : \underset{\text{domain}}{A} \longrightarrow \underset{\text{co-domain}}{B}$$

$\text{im}(f) = f(A) := \bigcup_{a \in A} \{f(a)\}$ ← the image of f ; $\text{im}(f) \subseteq B$
aka, the range

$f^{-1}(B') = \{ a \in A \mid \exists b \in B' \text{ s.t. } f(a) = b \}$

↑ the preimage of $B' \subseteq B$

e.g.,



$f(A) = \bigcup_{a \in A} \{f(a)\} = \{x\} \cup \{x\} = \{x\} \subseteq B$

In groups: 1 Selection Sort (outer loop)

2 Bubble Sort (outer loop)

In both, the outer loop goes from $i = 1$ to $n-1$

So, our decrementing fcn looks like this:

$$f : S \to \mathbb{N}$$

$$* \longmapsto n - i$$

Note: S is the ~~set~~ state space of the program.

pause the execution. What are all of the variable values (either implicit or explicit)

state space = set of all possible realized states