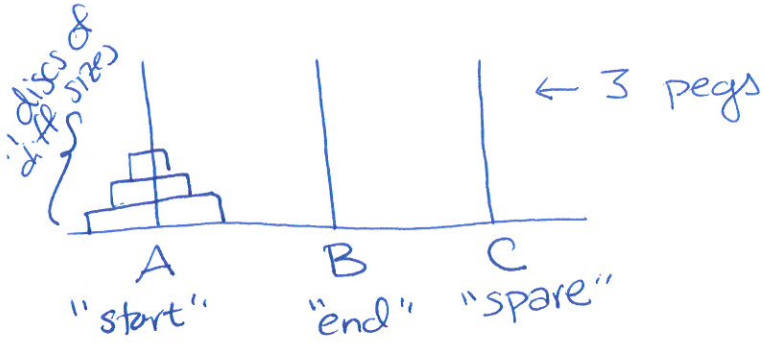
7 Sept 2022



Towers of Hanoi

WHAT

The problem: We have 3 pegs (A, B, and C) and n discs. All n discs are on peg A & we want them on peg B. Constraint: move 1 at a time + can never rest a larger disc on top of a smaller one.

HOW

The sol'n:

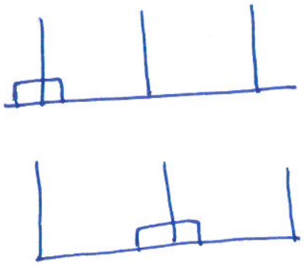
Hanoi(ndiscs, start, end, spare)

$\Theta(1)$	if ndiscs = 0, done!
$\Theta(1)$	else
$T(n-1)$	Hanoi(ndiscs − 1, start, spare, end)
$\Theta(1)$	move one disc from start to end
$T(n-1)$	Hanoi(ndiscs − 1, spare, end, start)
$\Theta(1)$	end ifelse

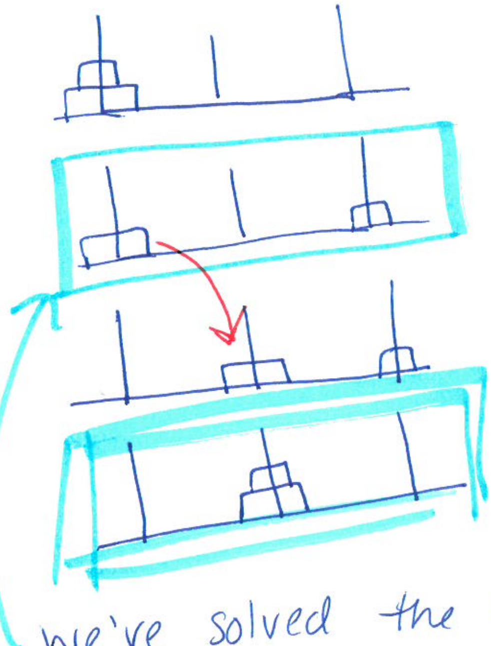
In HW, give both pseudocode and a prose description to satisfy "How"

Small examples: always great to start to get a sense!

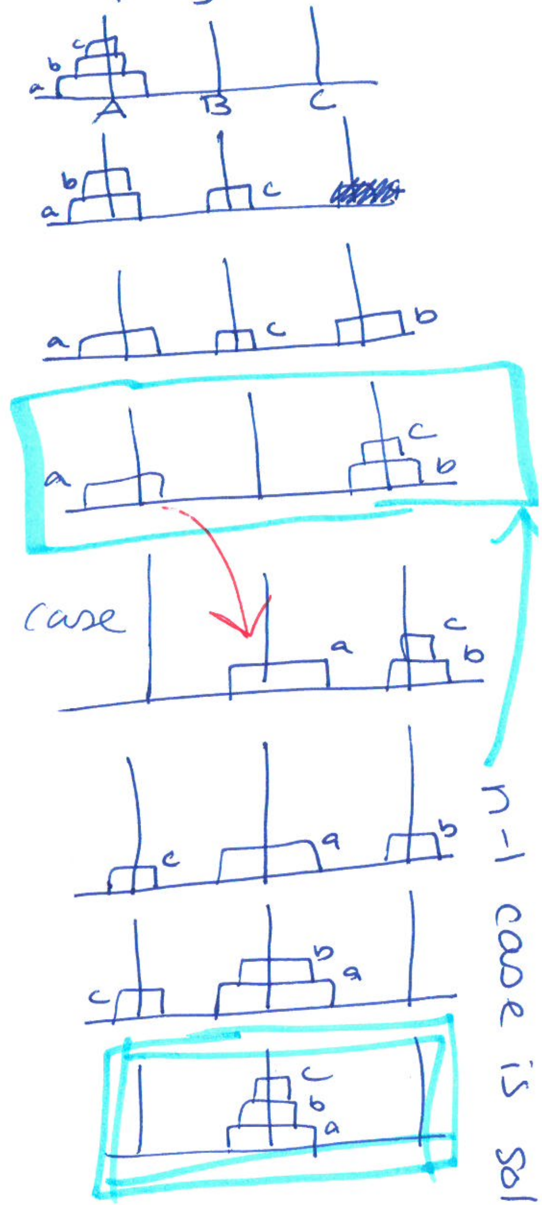
n = 1



n = 2



n = 3



We've solved the n-1 case
start = A
end = C
spare = B

n-1 case is solved!

Recursion: Solving a problem by reducing it to one (or more) ~~pro~~ same problems on smaller input.

trusting that the smaller cases are solved correctly is analagous to our inductive assumption in induction.

How FAST

How fast does Hanoi (n, A, B, C) take?

Let $T(n)$ = time to run Hanoi with ndiscs = n.

adding the cost of each line:

$$T(n) = \Theta(1) + \Theta(1) + T(n-1) + \Theta(1) + T(n-1) + \Theta(1)$$

$$= 2T(n-1) + 4\Theta(1)$$

$$= 2T(n-1) + \Theta(1)$$

~~T(n-1)~~

$$\boxed{T(n) = 2T(n-1) + \Theta(1)}$$ ← recursive (asymptotic) formula.

recursive (nonasymptotic) formula

$$T'(n) = \begin{cases} 0 & , n=0 \\ 2T(n-1)+1, & \text{if } n \geq 1 \end{cases}$$

I just picked a constant to test things out

Claim: $T'(n) = 2^n - 1$ ← closed form of T'

Proof by induction!

$T(n)$ is $\Theta(2^n)$ ← asymptotic form

note: we make asymptotic form as simple as possible kSort

m: given unsor~~ted~~ed array of $\mathbb{R}$-valu, sort the elements from smallest largest

So:

(A)

$n \leftarrow |A|$ (or $n \leftarrow$ A.size)

$k \leftarrow$ randint$(1, n)$ this is our pivot!

$S \leftarrow$ array of ets $< k$

$K \leftarrow$ array of elt~~s~~ $= k$

$L \leftarrow$ array of et~~s~~ $> k$

$S \leftarrow QS(S)$

$L \leftarrow QS(L)$

eturn $S \| K \| L$

concatenation of arrays

Towers of Hanoi

$$\left.\begin{array}{l} T(0)=0 \\ T(n) = 2T(n-1)+1 \end{array}\right\} T(n) = 2^n - 1$$

WHAT prob.
HOW soln
HOW FAST RT
WHY correctness

Merge Sort

$$T(n) = 2\,T(n/2) + \Theta(n)$$
$$= \Theta(n \log n)$$

Quicksort

→ always smallest $T(n) \leq T(n-1) + T(1) + \Theta(n) = O(n^2)$

→ always largest $T(n) = 2T(n/2) + \Theta(n)$