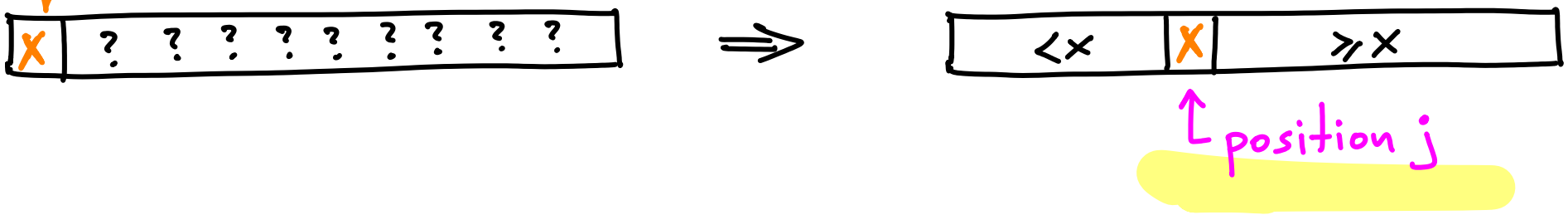
QUICKSORT

An in-place divide & conquer algorithm

- DIVIDE: choose a pivot & partition → smaller values go left, larger values go right



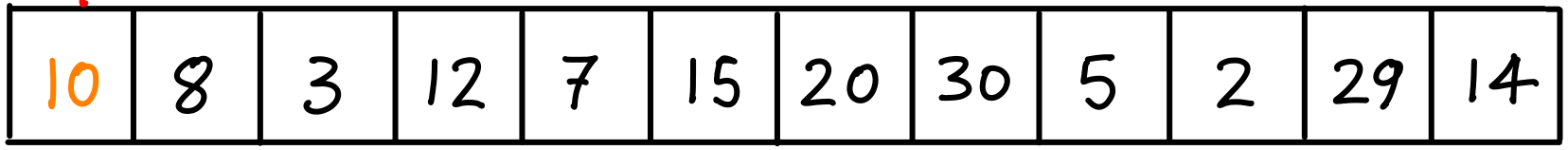
- CONQUER: Quicksort each side

Unlike Mergesort, there is no Combine phase.

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

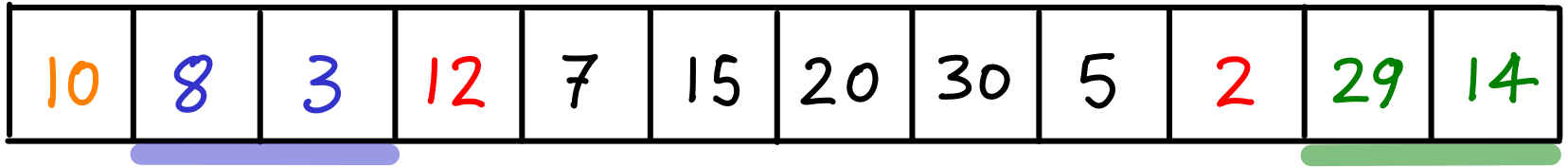
- DIVIDE: choose a pivot & partition

arbitrarily choose first

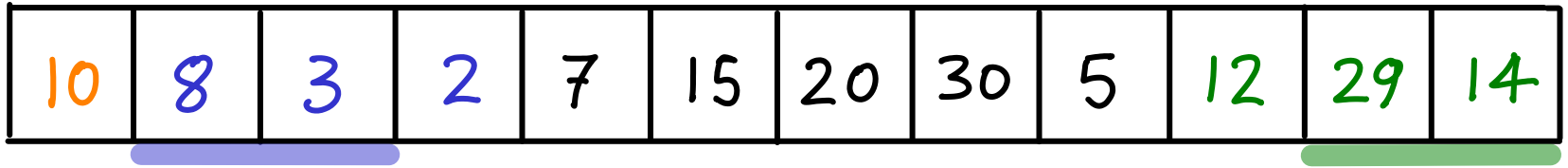


Grow "prefix" of smaller elements

Grow suffix of larger elements



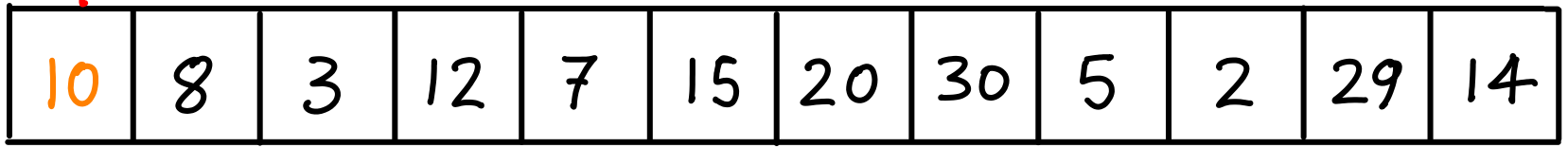
Now either the two sides meet or we can SWAP



... continue

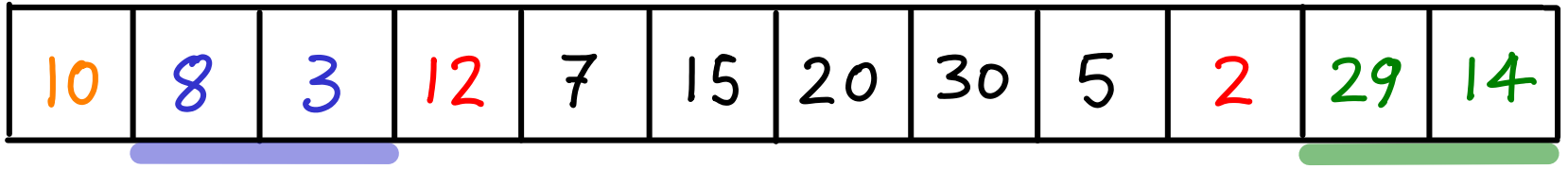
- DIVIDE: choose a pivot & partition

arbitrarily choose first

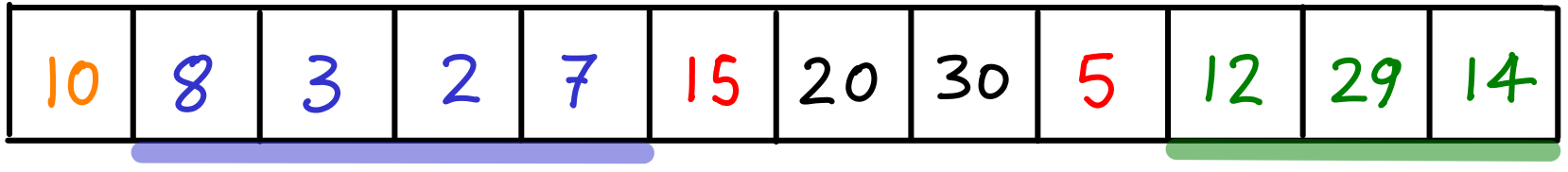


Grow "prefix" of smaller elements

Grow suffix of larger elements



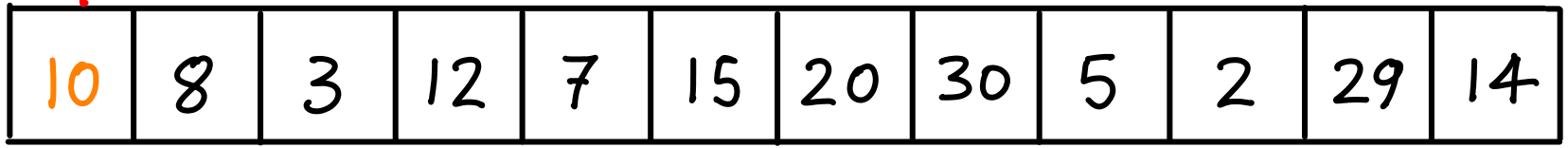
Now either the two sides meet or we can SWAP



... continue

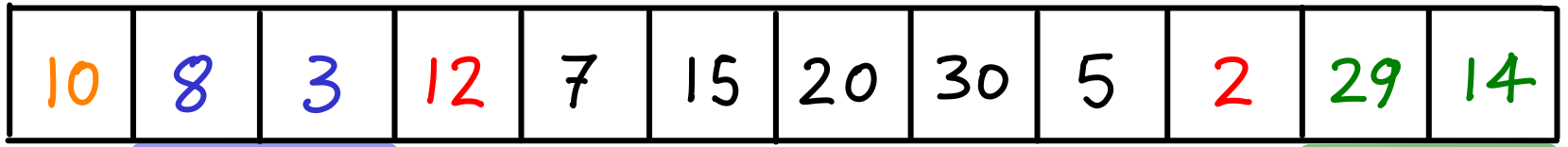
- DIVIDE: choose a pivot & partition

arbitrarily choose first

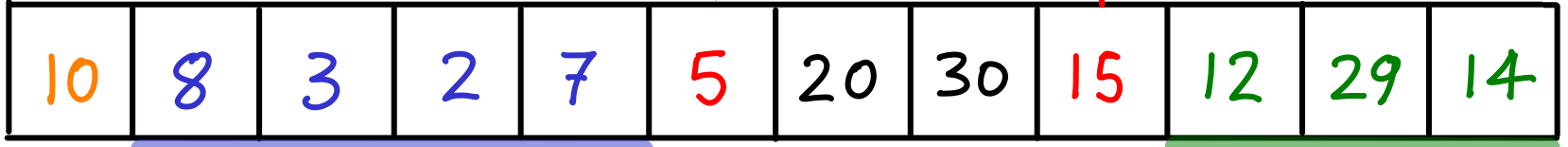


Grow "prefix" of smaller elements

Grow suffix of larger elements



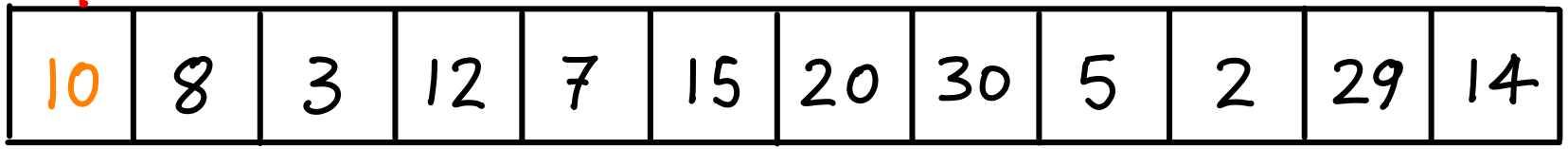
Now either the two sides meet or we can SWAP



... continue

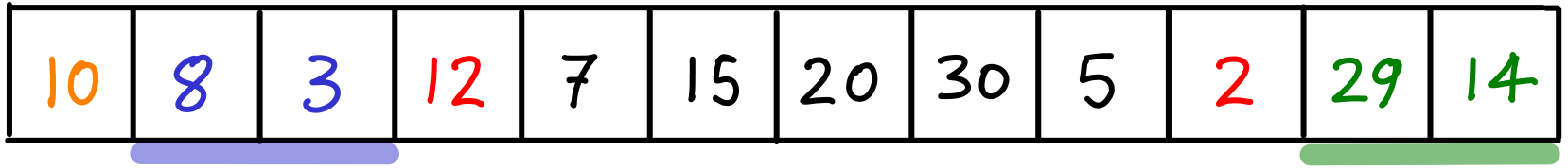
- DIVIDE: choose a pivot & partition

arbitrarily choose first

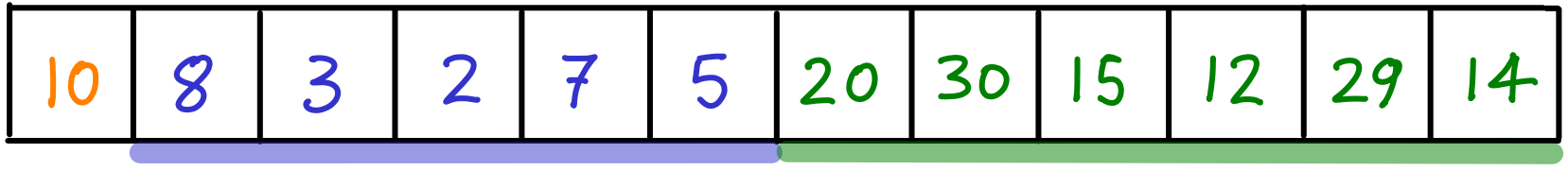


Grow "prefix" of smaller elements

Grow suffix of larger elements

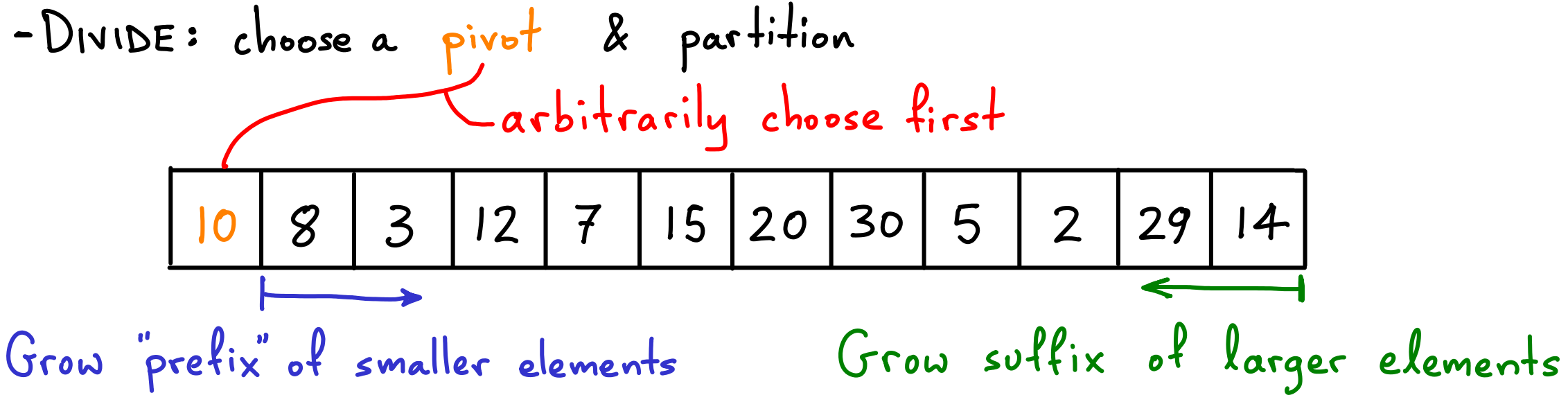


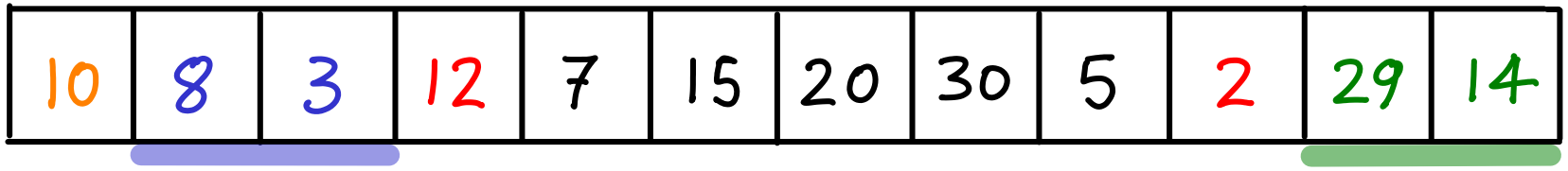
Now either the two sides meet or we can SWAP



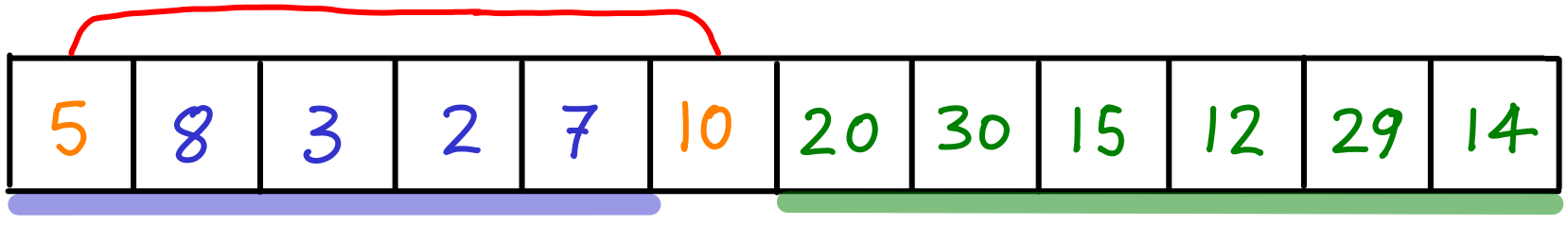
...continue

- DIVIDE: choose a pivot & partition

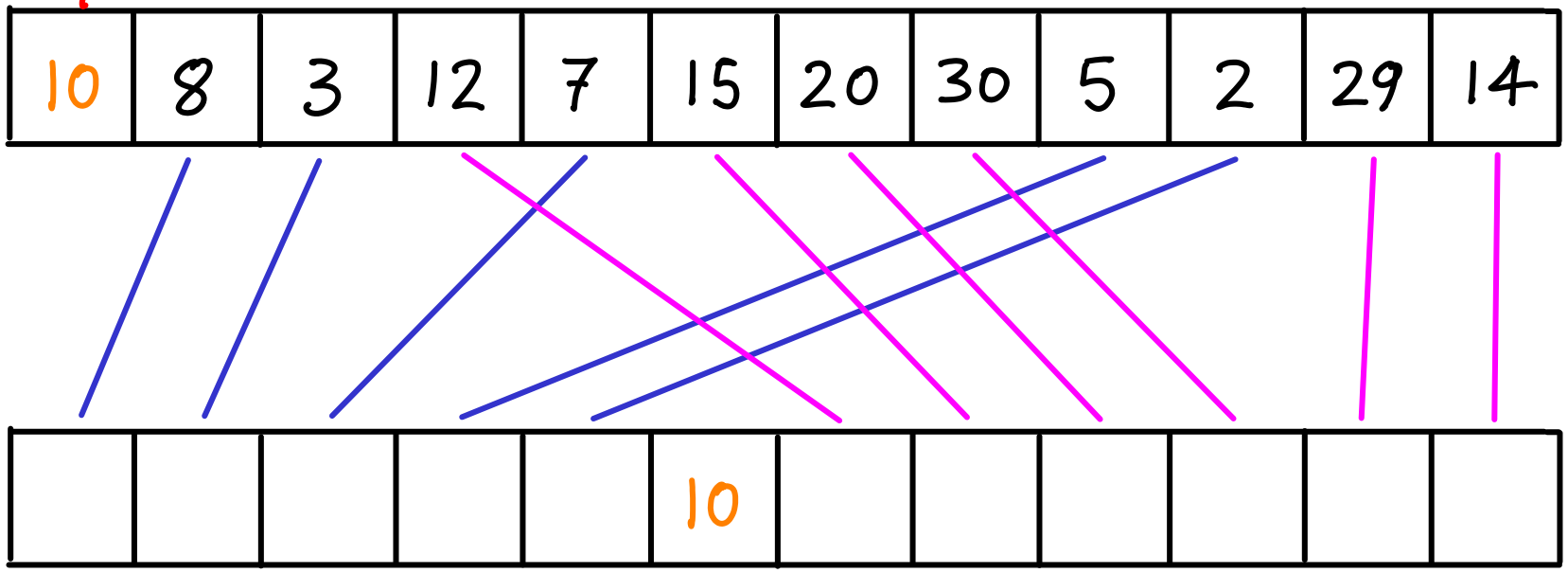




Now either the two sides meet or we can SWAP



- DIVIDE: choose a pivot & partition

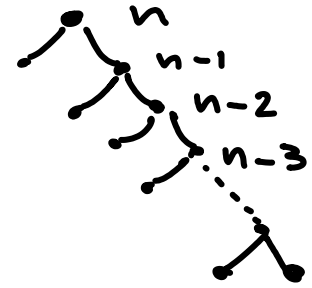


There are stable versions as well

What is the worst-case time complexity, and why?

↳ already sorted input, reverse-sorted, nearly sorted...

$T(n) = T(0) + T(n-1) + \Theta(n) = \Theta(n^2)$



What would be ideal ?

↳ ~ even split, every time

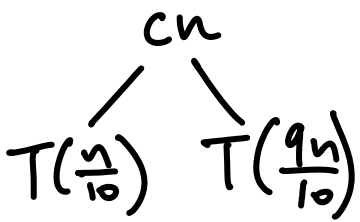
$T(n) \leq 2 \cdot T(\frac{n}{2}) + \Theta(n) = \Theta(n \log n)$

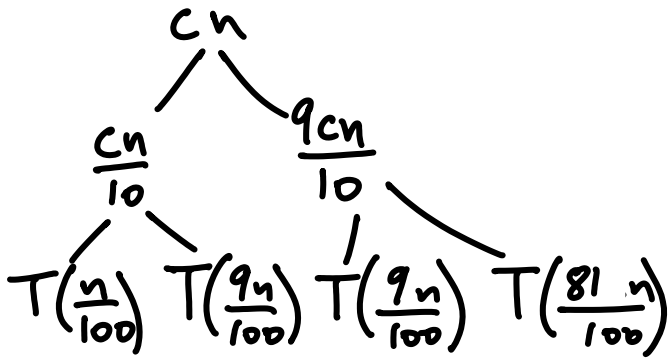
Why use Quicksort? We expect $\Theta(n \log n)$ ... with a small constant

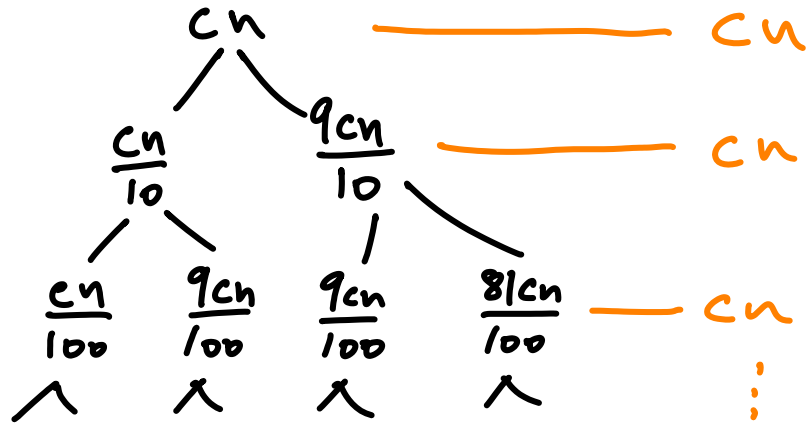
(& it's in-place, and stable)

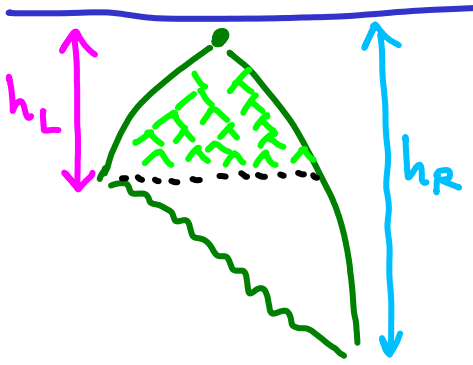
What if we always split "sort-of-evenly"?

e.g., $T(n) = T(\frac{n}{10}) + T(\frac{9n}{10}) + c \cdot n$









$h_L \sim \log_{10} n \rightarrow T(n) \geq cn \log_{10} n$

$h_R \sim \log_{10/9} n \rightarrow T(n) \leq cn \log_{10/9} n$

Any constant-fraction-split will give $\Theta(n \log n)$

What if we alternate between ideal & bad splits? (Lucky vs Unlucky)

$$\left.\begin{aligned} L(n) &= 2U\left(\tfrac{n}{2}\right) + dn \\ U(n) &= L(n-1) + dn \end{aligned}\right\}$$

$$L(n) = 2 \cdot \left[L\left(\tfrac{n}{2} - 1\right) + d\tfrac{n}{2}\right] + dn$$

$$\leq 2 \cdot L\left(\tfrac{n}{2}\right) + 2dn = \Theta(n \log n)$$

Expected time: call a split balanced if pivot ranks in $[\frac{n}{4} \ldots \frac{3n}{4}]$
unbalanced otherwise

Worst case if balanced split: $T(n) \leq T(\frac{3n}{4}) + T(\frac{n}{4}) + dn$

Worst case if unbalanced split: $T(n) \leq T(n) + dn$

Each split has a 50% chance of being balanced

$$T(n) \leq 0.5(T(n)+dn) + 0.5 \cdot (T(\tfrac{3n}{4}) + T(\tfrac{n}{4}) + dn)$$

$$0.5\,T(n) \leq dn + 0.5(T(\tfrac{3n}{4}) + T(\tfrac{n}{4}))$$

$$T(n) \leq T(\tfrac{3n}{4}) + T(\tfrac{n}{4}) + 2dn = \Theta(n \log n)$$