BACK TO SORTING: QUICKSORT

a Divide & Conquer algorithm that runs “in-place”

- Divide: choose a pivot & place it s.t. everything before is smaller & everything after is not smaller

\[ \begin{array}{cccccccccccccc} \text{X} & \text{?} & \text{?} & \text{?} & \text{?} & \text{?} & \text{?} & \text{?} & \text{?} & \text{?} & \text{?} & \text{?} & \text{?} \end{array} \Rightarrow \begin{array}{cccccccccccccc} \text{<x} & \text{X} & \text{>x} \end{array} \]

- Conquer: Quicksort each side of pivot

* Notice that after Divide, X is in its final (sorted) position. So there is nothing to do for [Combine]

\[ T(n) = T(j-1) + T(n-j) + f(n) \quad \text{//} \quad f(n) = [\text{Divide n}] \]
\[ T(0) = 0 \]
\[ T(1) = \Theta(1) \]
The heart of Quicksort is the **Divide** step.

- **pivot**: arbitrary → so just use first element
- **in-place** indexing of pivot:

  - non-trivial case: you have at least one element < x
    - non-empty
      - ? → if all ≥ x, done
    - possibly empty

  - Scan from z: if z ≥ x, advance
    - else swap z ↔ a
      - if it exists

  - if all these are ≥ x, nothing to do.
  - So advance until you find y < x
    - (or until the end)
    - e.g. find y immediately
    - or, you don’t find y immediately

  - swap y for the element to the right of x
  - advance in array
We said Quicksort is fast, i.e. \( \Theta(n \log n) \), if pivot gives an even split. (ALWAYS)

What if we always split within \( \frac{1}{10} \) to \( \frac{9}{10} \)?

\[
T(n) = T\left(\frac{n}{10}\right) + T\left(\frac{9n}{10}\right) + cn.
\]

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

\( h_L \sim \log_{10} n \Rightarrow T(n) \sim cn \cdot \log_{10} n \)

\( h_R \sim \log_{10/9} n \Rightarrow T(n) \leq cn \cdot \log_{10/9} n \)
Repeat: if you could ensure that every pivot gives some [constant fraction of n]-split, you would get \( \Theta(n \log n) \)

...possibly with terrible hidden constant

we can’t do that, but it might happen with high probability.

Let’s look at another example: alternate balanced & unbalanced split.

\[
\begin{align*}
L(n) &= 2U\left(\frac{n}{2}\right) + \Theta(n) \quad \text{lucky} \\
U(n) &= L(n-1) + \Theta(n) \quad \text{unlucky}
\end{align*}
\]

\[
L(n) = 2\left[L\left(\frac{n}{2}-1\right) + \Theta\left(\frac{n}{2}\right)\right] + \Theta(n) = 2L\left(\frac{n}{2}-1\right) + \Theta(n) = \Theta(n \log n)
\]

A note: we can avoid specific “bad” distributions by permuting the input or random pivot selection.

If all input permutations are equally likely, then picking the first element is fine.