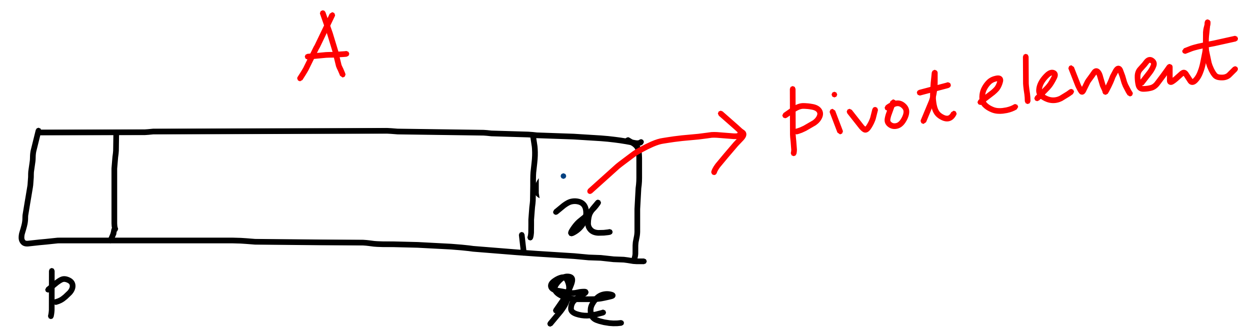
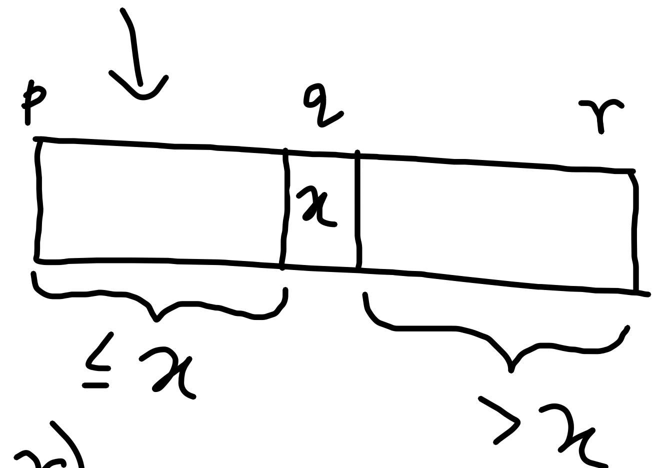
# Sorting

- Insertion Sort
- Selection Sort
- Merge Sort
- Heap Sort
- Quick Sort




## Divide & Conquer

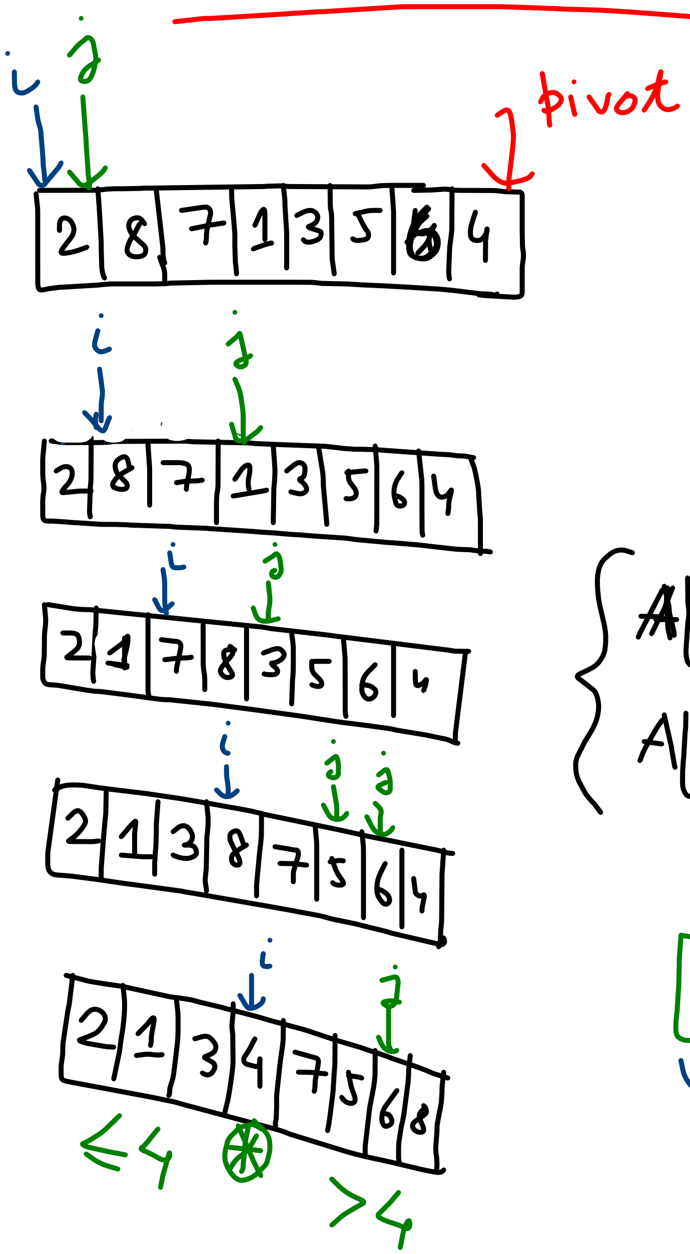
QSort (A, p, r)

$q \leftarrow$ Partition (A, p, r)

QSort (A, p, q-1)

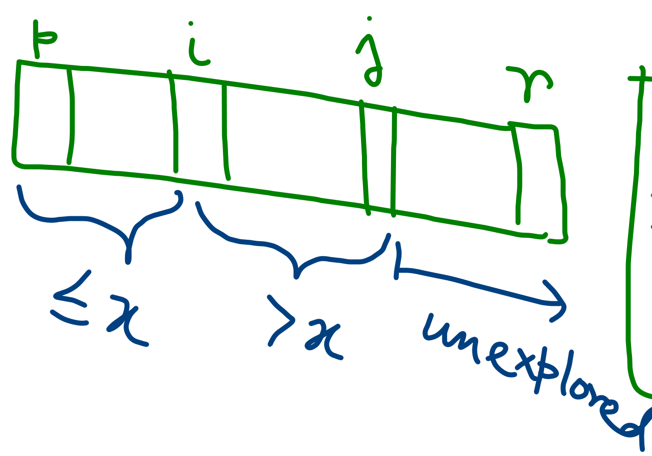
QSort (A, q+1, r)

# Partitioning An Array




If $A[j] \le$ pivot
 swap$(A[i], A[j])$
 $i{+}{+}$

$$\begin{cases} A[1..(i-1)] \le \text{pivot} \\ A[i \;..(j-1)] > \text{pivot} \end{cases}$$




Loop Invariant

## Partition $(A, p, r)$

pivot $= A[r]$
$i = p-1$
⇒ For $j = p\,(1)\,r-1$
 if $A[j] \le$ pivot
  $i = i+1$
  Swap$(A[i], A[j])$
Swap$(A[i+1], A[r])$
return $(i+1)$

# Time Complexity

**Worst Case:** $T(n) = T(i) + T(n-i-1) + O(n)$

$i = n-1$ $\Rightarrow$ $T(n) = T(n-1) + O(n)$

$= T(n-2) + O(n-1) + O(n)$

$= \ldots$

$O(n^2)$

**Best Case:**

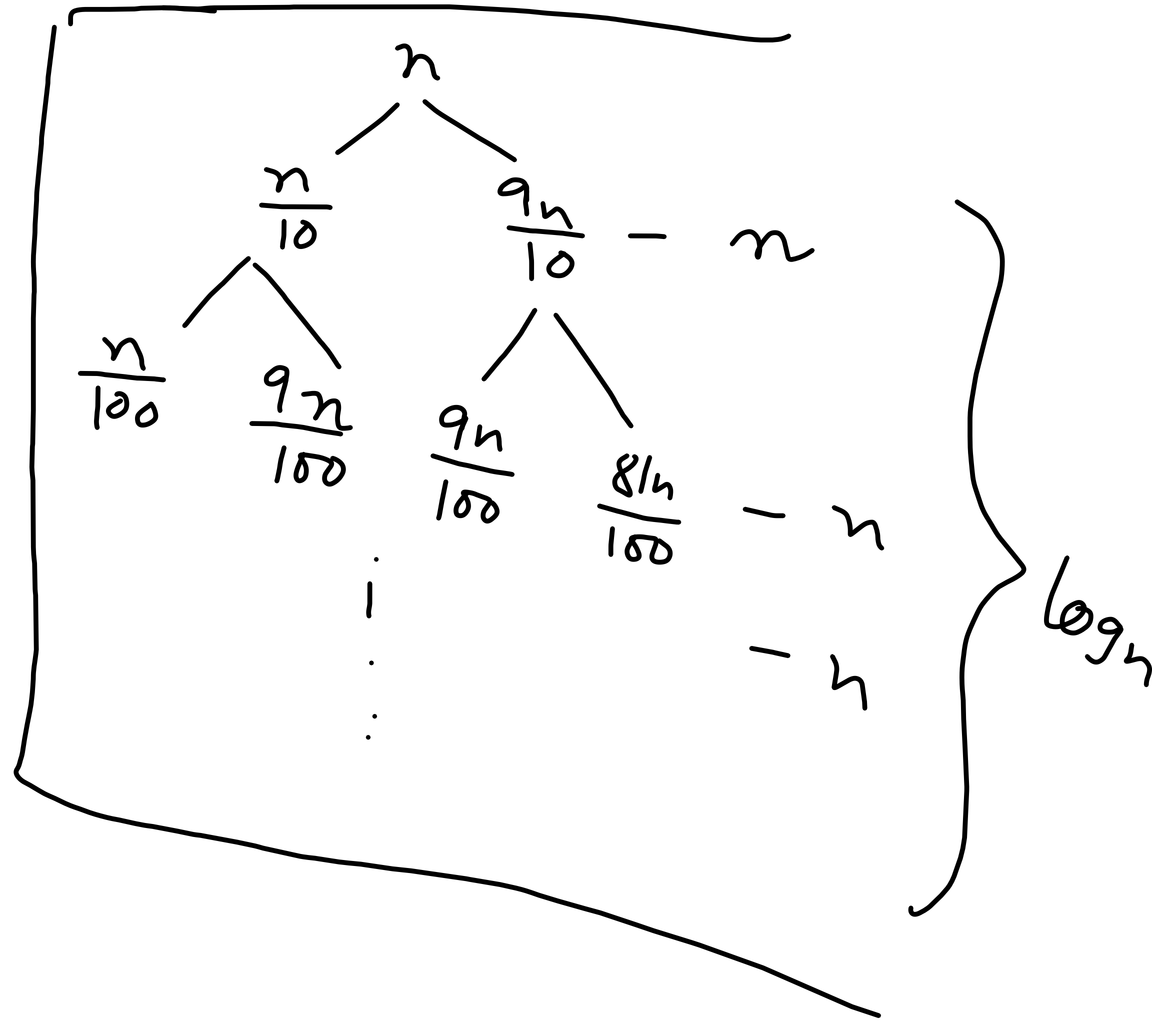
$i = n/2$ $\Rightarrow$ $T(n) = 2T(n/2) + O(n)$

$= O(n \log n)$

# Avg Case (Intuition)

$$T(n) = T\left(\frac{n}{10}\right) + T\left(\frac{9n}{10}\right) + O(n)$$

$$\approx \Theta(n \log n)$$




## Randomized Quick Sort

- Choose $i \xleftarrow{\$} \{p, \dots, r\}$
- Swap$(A[i], A[r])$
- Use QuickSort$(A, p, r)$

# Time Complexity

**Worst Case:** $T(n) = \max_{0 \le q \le n-1} \big(T(q) + T(n-q+1)\big) + \Theta(n)$

$= \Theta(n^2)$

**Avg Case:**
(Expected time complexity)

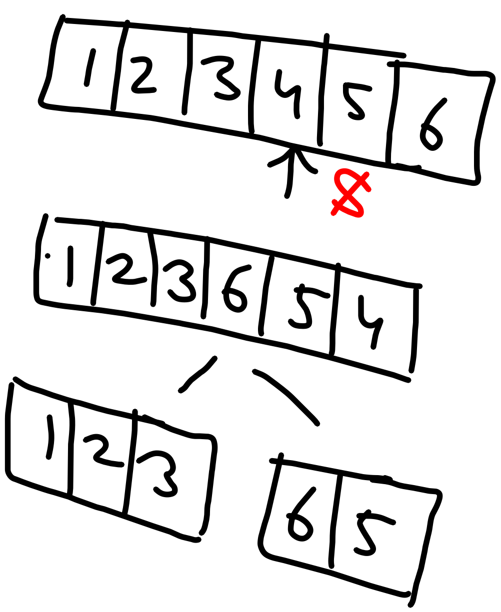
- $X \rightarrow$ # comparisons in the whole algo
- When two elements are compared?

$A = \{z_1, \ldots, z_n\}$ $z_i \rightarrow i^{th}$ min element

$Z_{ij} = \{z_i, \ldots, z_j\}$

When $z_i$ & $z_j$ are compared?

$\hookrightarrow$ $\boxed{z_i \text{ or } z_j \text{ is pivot of } Z_{ij}}$




No comparison between two elements where one is $\le$ pivot, other $>$ pivot

$$X_{ij} = I_{\{z_i \text{ is compared with } z_j\}} \Rightarrow$$

$X_{ij} = 1$, if $z_i$ is compared with $z_j$

$= 0$, o/w

$$X = \sum_{i=1}^{n} \sum_{j=i+1}^{n} X_{ij}$$

$$E[X] = \sum_{i=1}^{n} \sum_{j=i+1}^{n} \Pr[z_i \text{ is compared with } z_j]$$

$$= \sum_{i=1}^{n} \sum_{j=i+1}^{n} \frac{2}{j-i+1} = O(n \log n)$$

$$= \sum_{i=1}^{n} \sum_{k=1}^{n} \frac{2}{k} \rightarrow O(\log n)$$

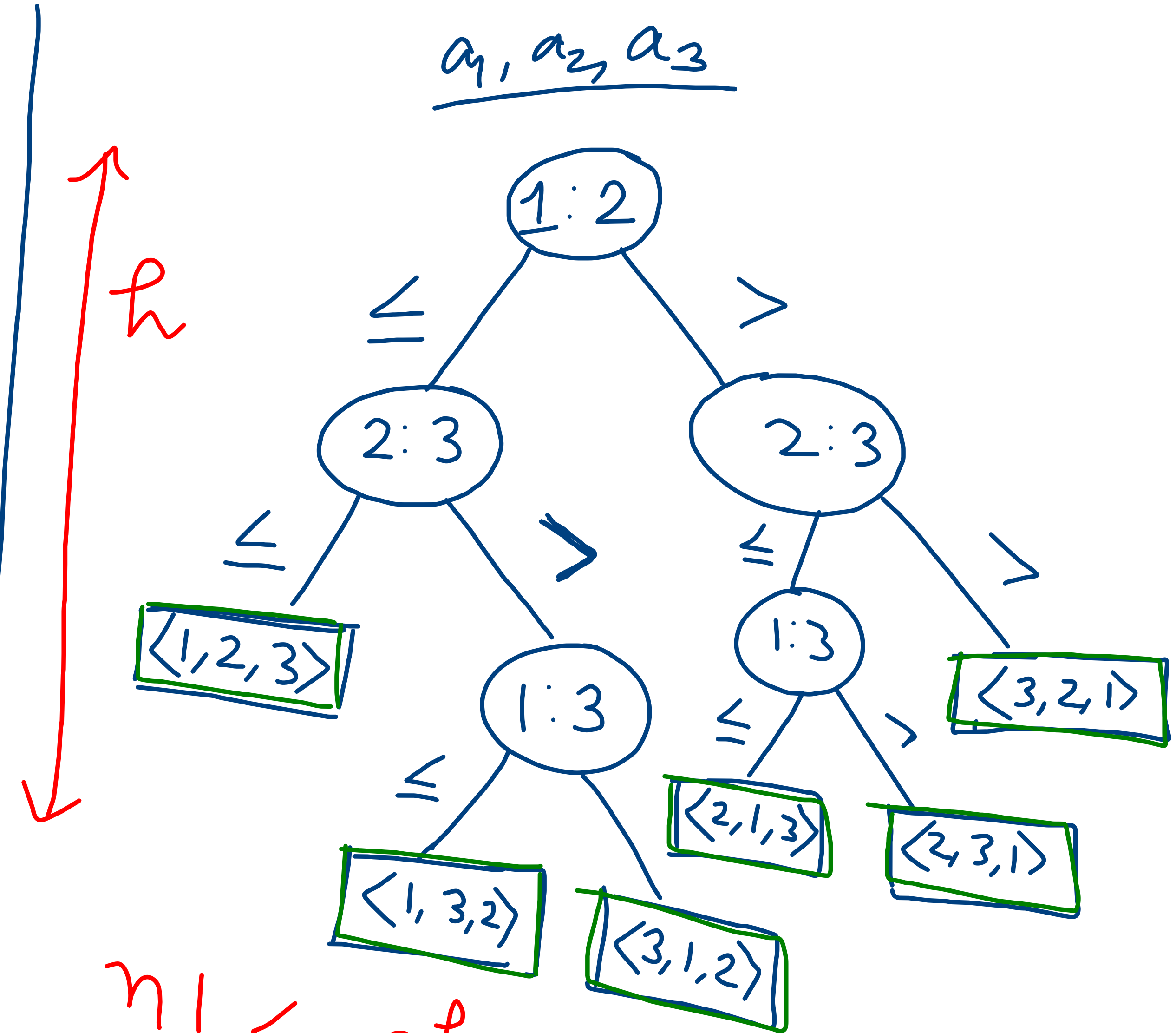
- Running Time → $\Theta(n \log n)$
- Can we do better?

## Comparison-based Sort

- Comparison between elements
- No restriction on inputs

## Decision-Tree

Full binary tree with leaves (each node 0/2 child) as sorted order




$$n! \leq 2^h$$

$$\Rightarrow h \simeq \Omega(n \log n)$$

→ Stirling approximation

$$n! = \sqrt{2\pi n} \left(\frac{n}{e}\right)^n e^{\alpha_n}$$

# Linear Sort