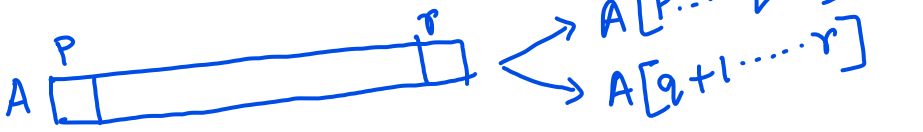
Divide and Conquer:

1. **Divide** the problem into a number of subproblems that are smaller instances of the same problem.
2. **Conquer** the subproblems by solving them recursively. If the subproblem sizes are small enough, however, just solve the subproblems in a straightforward manner.
3. **Combine** the solutions to the subproblems into the solution for the original problem.



Quicksort:

Divide: Partition (rearrange) the array $A[p..r]$ into two (possibly empty) subarrays $A[p..q-1]$ and $A[q+1..r]$ such that each element of $A[p..q-1]$ is less than or equal to $A[q]$, which is, in turn, less than or equal to each element of $A[q+1..r]$. Compute the index q as part of this partitioning procedure.

Conquer: Sort the two subarrays $A[p..q-1]$ and $A[q+1..r]$ by recursive calls to quicksort.

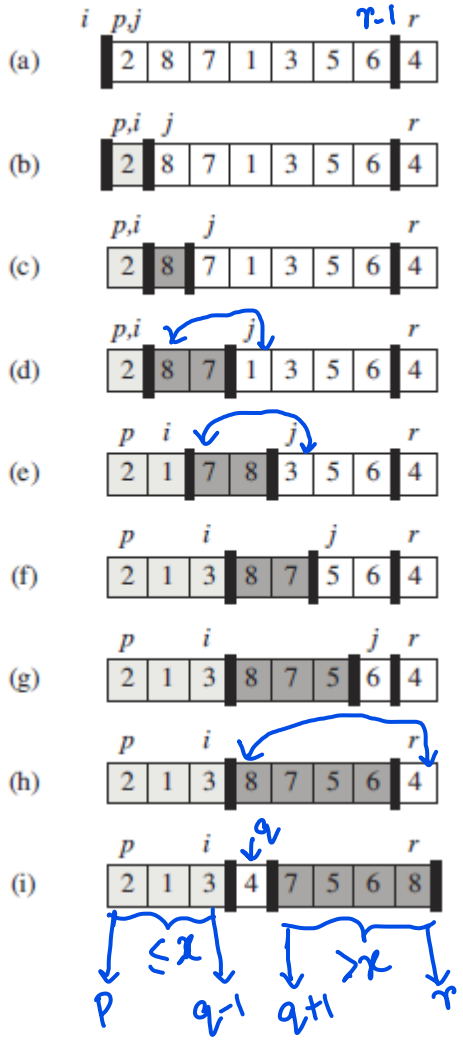
Combine: Because the subarrays are already sorted, no work is needed to combine them: the entire array $A[p..r]$ is now sorted.

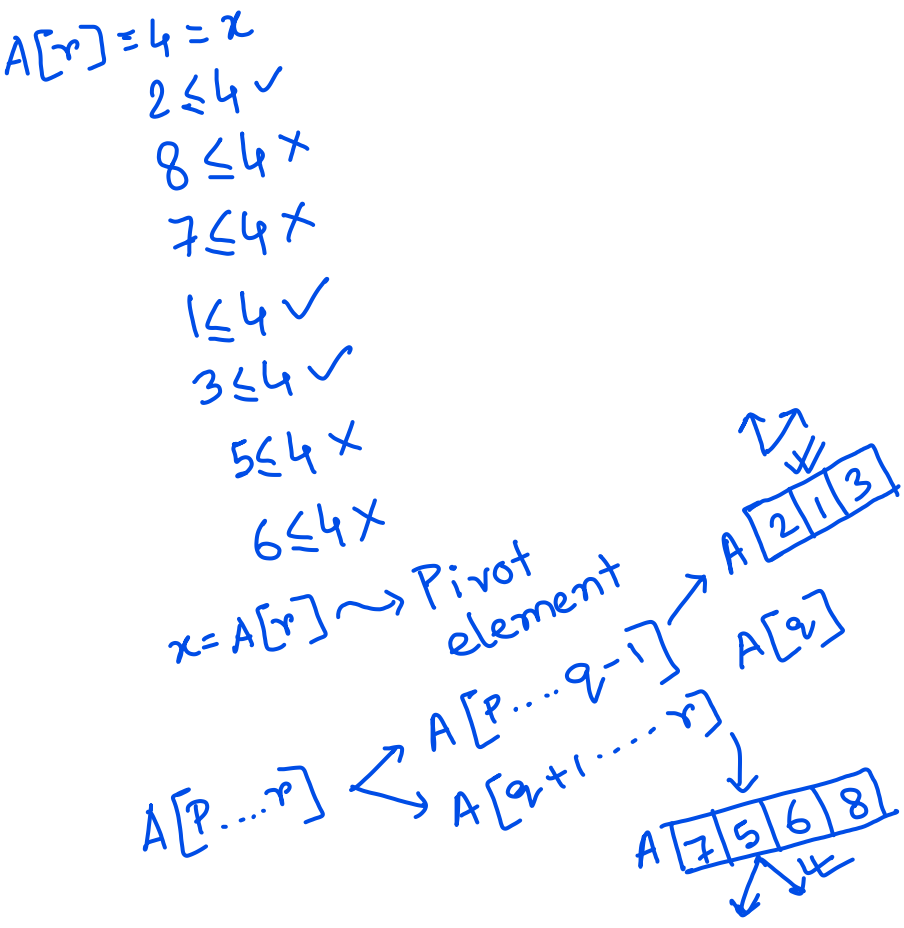
QUICKSORT(A, p, r)

```
1  if p < r
2      q = PARTITION(A, p, r)
3      QUICKSORT(A, p, q − 1)
4      QUICKSORT(A, q + 1, r)
```

PARTITION(A, p, r)

```
1  x = A[r]
2  i = p − 1
3  for j = p to r − 1
4      if A[j] ≤ x
5          i = i + 1
6          exchange A[i] with A[j]
7  exchange A[i + 1] with A[r]
8  return i + 1
```





Time complexity :

Time req. for partition $n-1$ elements, 0 element

Worst case :

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

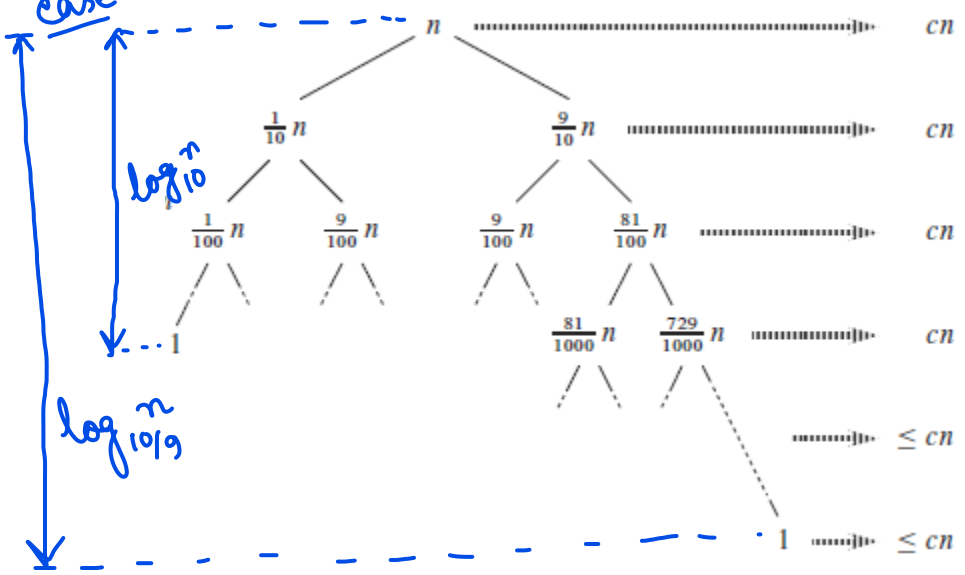
$\rightarrow$ Const. time

$\rightarrow \Theta(n^2)$

Best case : $T(n) = 2T(n/2) + \Theta(n)$ $\rightarrow \Theta(n \log n)$

Average case : $T(n) = T(9n/10) + T(n/10) + cn$

9:1 proportion



$n \log_{10/9} n$