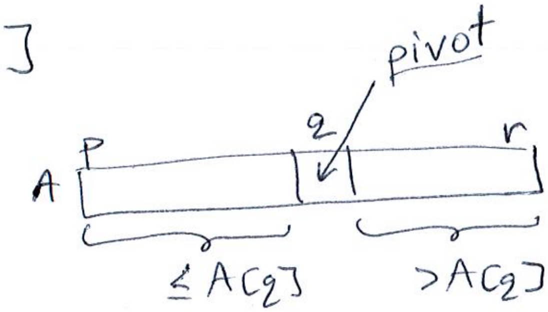
# Quicksort

- worst-case RT is $\Theta(n^2)$
- expected RT is $\Theta(n \lg n)$
- runs very fast in practice
- Insertion Sort - sorting in place / a constant # of elements are stored outside of array at any given time.
- uses divide-and-conquer approach

## Divide-and-conquer

- general problem: sort $A[p..r]$

**Divide**:




partition the array $A[p..r]$ into two subarrays $A[p..q-1]$ and $A[q+1..r]$ such that:

- elements in $A[p..q-1]$ are $\leq A[q]$
- elements in $A[q+1..r]$ are $> A[q]$

Index $q$ is computed at this step.

**Conquer**

recursively sort $A[p..q-1]$ and $A[q+1..r]$

**Combine**

nothing to be done

## QUICKSORT(A, p, r)

```
if p<r
    q = PARTITION(A, p, r)        // divide
    QUICKSORT(A, p, q-1)          // conquer
    QUICKSORT(A, q+1, r)          // conquer
```

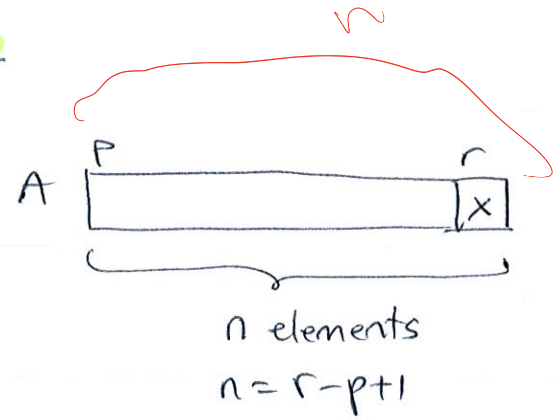
$A, p, length)$

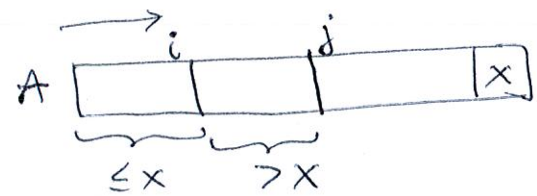
- initial call is QUICKSORT$(A, 1, A.length)$

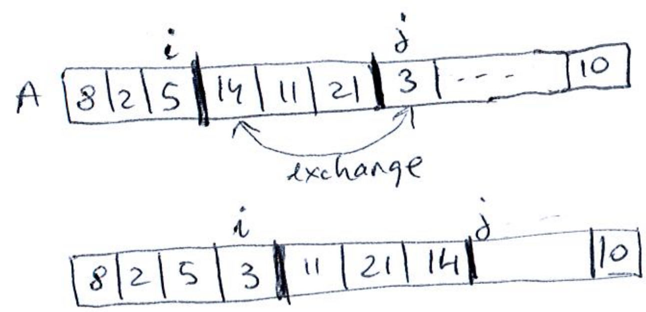
## Partitioning the array (divide step)

```
PARTITION(A,p,r)
x = A[r]
i = p-1
for j = p to r-1
    if A[j] ≤ x
        i = i+1
        exchange A[i] with A[j]
exchange A[i+1] with A[R]
return i+1
```

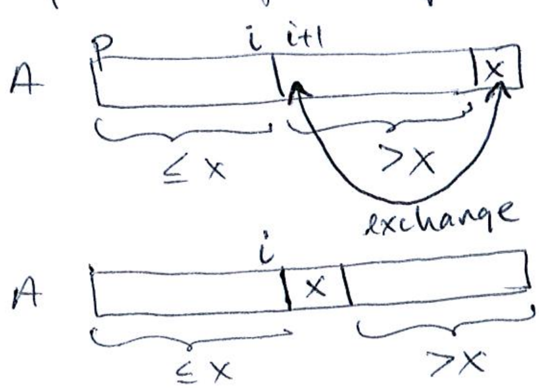
$\Theta(n)$




- after the for loop:




### RT analysis

let $n = r-p+1$

$\boxed{RT = \Theta(n)} \longrightarrow$ RT for PARTITION

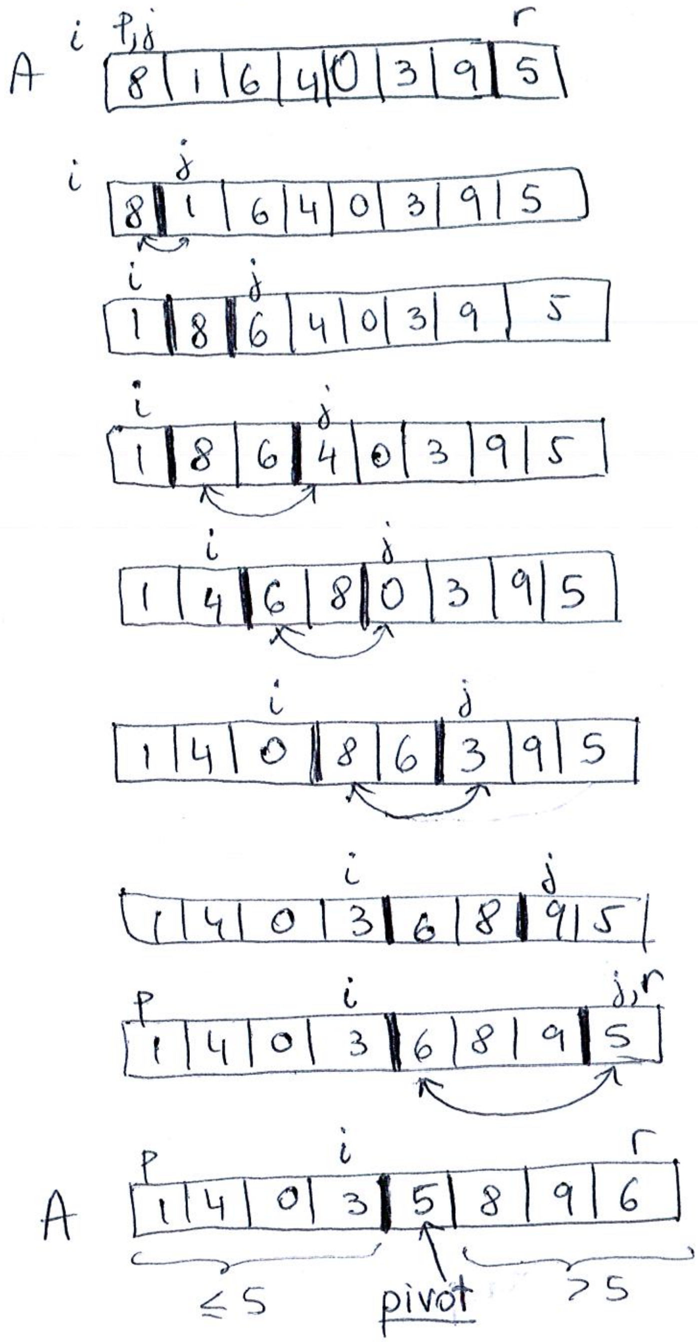
### Correctness

Loop invariant: at the start of each iteration $j$ of the for loop:

- elements in $A[p..i]$ are $\leq x$
- elements in $A[i+1..j-1]$ are $> x$
- $x = A[r]$ is the pivot

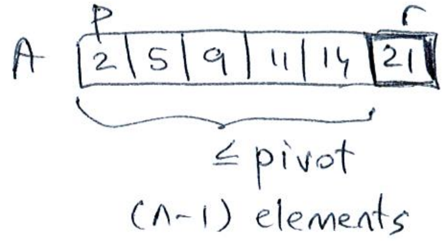
example of the operation PARTITION

last Element is pivot
x = 5 is the pivot




## RT analysis of quicksort

- Worst-case partitioning
  - the two subproblems are completely unbalanced:
    - → one subproblem has $(n-1)$ elements
    - → one subproblem has 0 elements
  - occurs when the input array is already sorted in increasing order




pivot = 21

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

↑ RT for PARTITION

$$T(n) = T(n-1) + cn$$

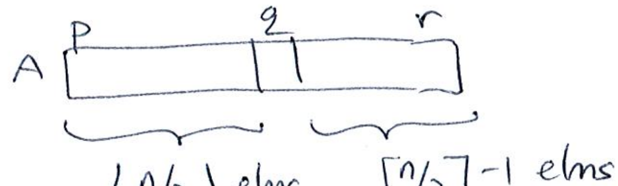
solved previously using Recursion Tree & Substitution method

$$\boxed{T(n) = \Theta(n^2)}$$

- Best-case partitioning
  - the two subproblems are completely balanced
  - for example one has size $\lfloor n/2 \rfloor$, and one of size $\lceil n/2 \rceil - 1$

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + \Theta(n)$$

↑ RT for PARTITION




$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + cn$$

$$\boxed{T(n) = \Theta(n \cdot \lg n)}$$ (Master Theorem, case 2)

Use Quicksort to sort the array A = <E, X, E, R, C, I, S, E> in alphabetical order. Draw a tree which shows the recursive calls made by Quicksort.

Answer: