ASSUME THAT YOU ARE GIVEN AN ARRAY AND AN ELEMENT IN THAT ARRAY. YOU HAVE ARRANGE THE ARRAY AS FOLLOWS.

PIVOT → 4 3 5 7 9 1 2 8 6

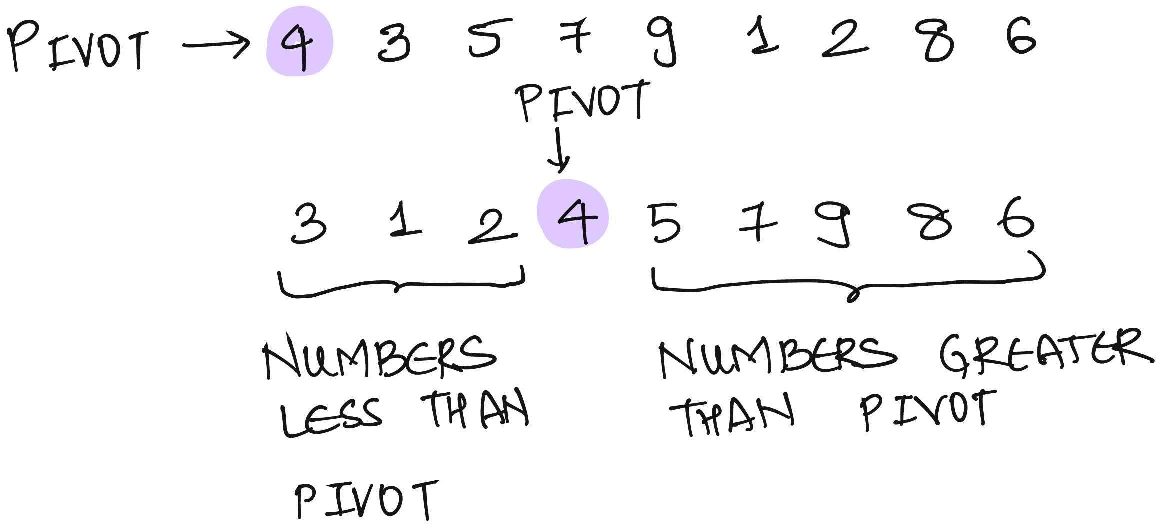
ASSUME THAT YOU ARE GIVEN AN ARRAY AND AN ELEMENT IN THAT ARRAY. YOU HAVE ARRANGE THE ARRAY AS FOLLOWS.

PIVOT → 4 3 5 7 9 1 2 8 6

PIVOT ↓

$$\underbrace{3\quad 1\quad 2}_{\text{NUMBERS LESS THAN PIVOT}}\quad 4\quad \underbrace{5\quad 7\quad 9\quad 8\quad 6}_{\text{NUMBERS GREATER THAN PIVOT}}$$

ASSUME THAT YOU ARE GIVEN AN ARRAY AND AN ELEMENT IN THAT ARRAY. YOU HAVE ARRANGE THE ARRAY AS FOLLOWS.



Q HOW WILL YOU ACCOMPLISH THIS TASK?

ASSUME THAT YOU ARE GIVEN AN ARRAY AND AN ELEMENT IN THAT ARRAY. YOU HAVE ARRANGE THE ARRAY AS FOLLOWS.

PIVOT → 4 3 5 7 9 1 2 8 6

$$\underbrace{3\quad 1\quad 2}_{\text{NUMBERS LESS THAN PIVOT}}\quad \overset{\text{PIVOT}}{\overset{\downarrow}{4}}\quad \underbrace{5\quad 7\quad 9\quad 8\quad 6}_{\text{NUMBERS GREATER THAN PIVOT}}$$

Q HOW WILL YOU ACCOMPLISH THIS TASK?

4 3 5 7 9 1 2 8 6

$P_<$									

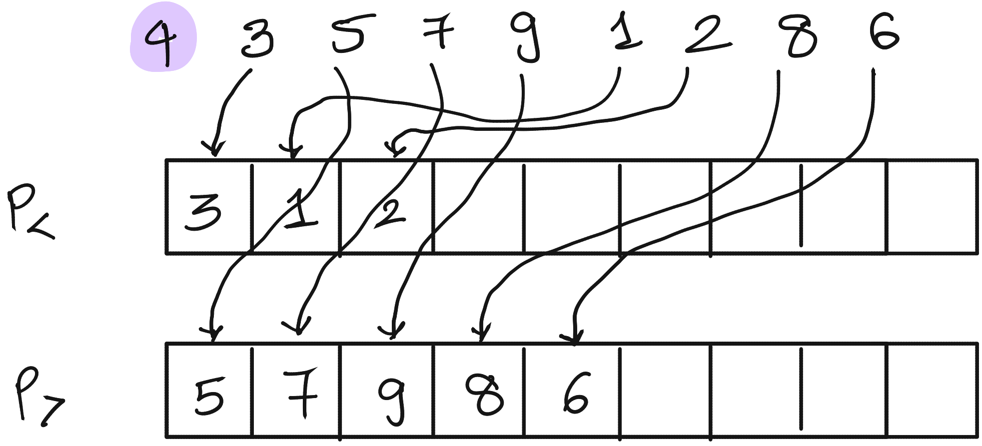
$P_>$									

ASSUME THAT YOU ARE GIVEN AN ARRAY AND AN ELEMENT IN THAT ARRAY. YOU HAVE ARRANGE THE ARRAY AS FOLLOWS.

PIVOT → 4 3 5 7 9 1 2 8 6

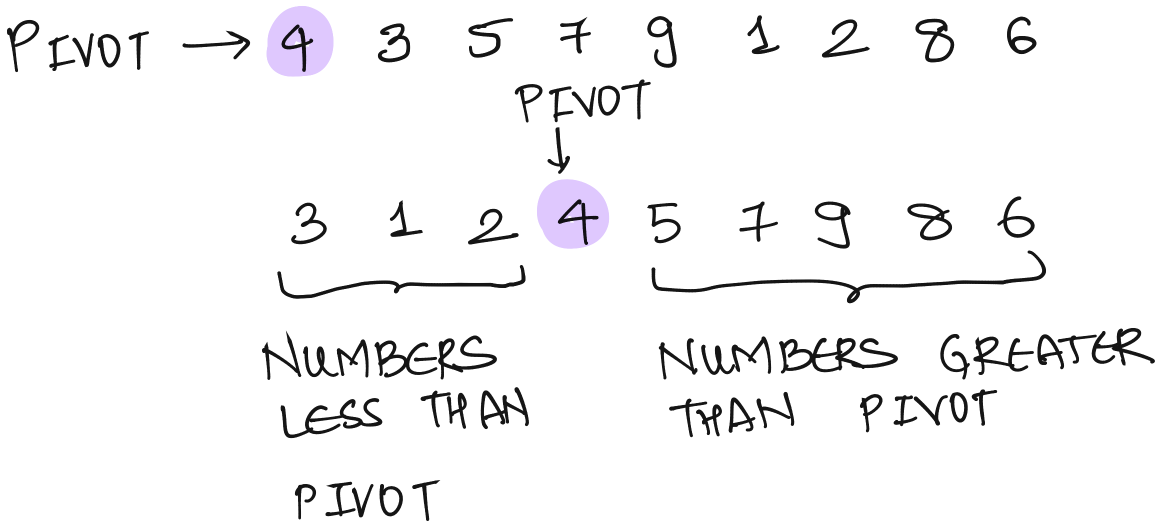
$$\underbrace{3\quad 1\quad 2}_{\text{NUMBERS LESS THAN PIVOT}}\quad \overset{\overset{\text{PIVOT}}{\downarrow}}{4}\quad \underbrace{5\quad 7\quad 9\quad 8\quad 6}_{\text{NUMBERS GREATER THAN PIVOT}}$$

Q HOW WILL YOU ACCOMPLISH THIS TASK?

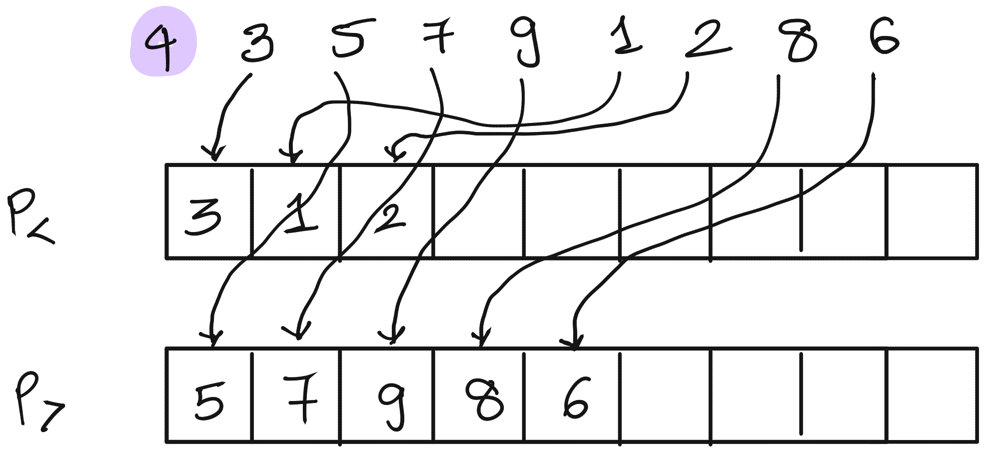


RUNNING TIME:

ASSUME THAT YOU ARE GIVEN AN ARRAY AND AN ELEMENT IN THAT ARRAY. YOU HAVE ARRANGE THE ARRAY AS FOLLOWS.



Q HOW WILL YOU ACCOMPLISH THIS TASK?



RUNNING TIME: $O(n)$.

ASSUME THAT YOU ARE GIVEN AN ARRAY AND AN ELEMENT IN THAT ARRAY. YOU HAVE ARRANGE THE ARRAY AS FOLLOWS.

PIVOT → 4 3 5 7 9 1 2 8 6

$$\underbrace{3 \quad 1 \quad 2}_{\text{NUMBERS LESS THAN PIVOT}} \quad \overset{\text{PIVOT}\ \downarrow}{4} \quad \underbrace{5 \quad 7 \quad 9 \quad 8 \quad 6}_{\text{NUMBERS GREATER THAN PIVOT}}$$

MAIN OBSERVATION: THE PIVOT (NUMBER 4) IS AT ITS CORRECT POSITION IN THE FINAL SORTED ARRAY.

ASSUME THAT YOU ARE GIVEN AN ARRAY AND AN ELEMENT IN THAT ARRAY. YOU HAVE ARRANGE THE ARRAY AS FOLLOWS.

PIVOT → 4 3 5 7 9 1 2 8 6

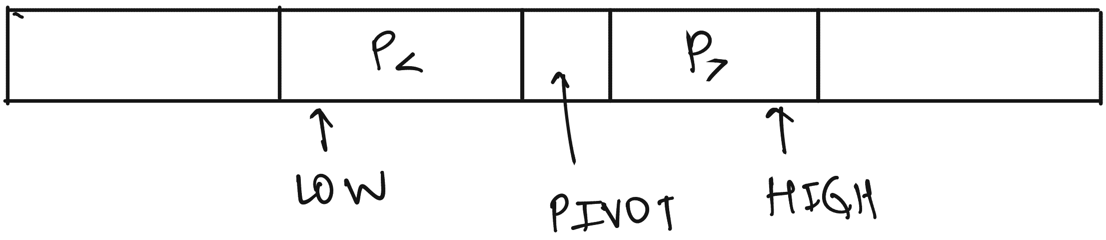
PIVOT
↓

$$\underbrace{3\ \ 1\ \ 2}_{\text{NUMBERS LESS THAN PIVOT}}\ \ 4\ \ \underbrace{5\ \ 7\ \ 9\ \ 8\ \ 6}_{\text{NUMBERS GREATER THAN PIVOT}}$$

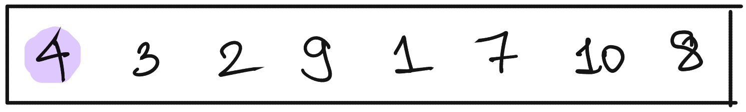
MAIN OBSERVATION: THE PIVOT (NUMBER 4) IS AT ITS CORRECT POSITION IN THE FINAL SORTED ARRAY.

RECURSE ON THE LEFT & RIGHT ARRAY.

```
QUICKSORT (A, LOW, HIGH)
{       IF (LOW = HIGH)
             RETURN;
        PIVOT ← A[LOW]
        $P_<$ ← NUMBERS LESS THAN PIVOT IN
                   A[LOW..... HIGH]
        $P_>$ ← NUMBERS GREATER THAN PIVOT IN
                   A[LOW .... HIGH]
```



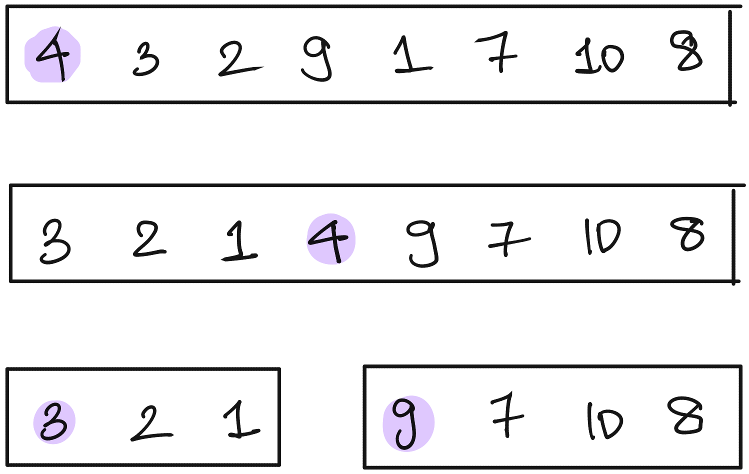
```
        QUICKSORT (A, LOW, LOW + $|P_<|$ - 1);
        QUICKSORT (A, LOW + $|P_<|$ + 1, HIGH);
}
```

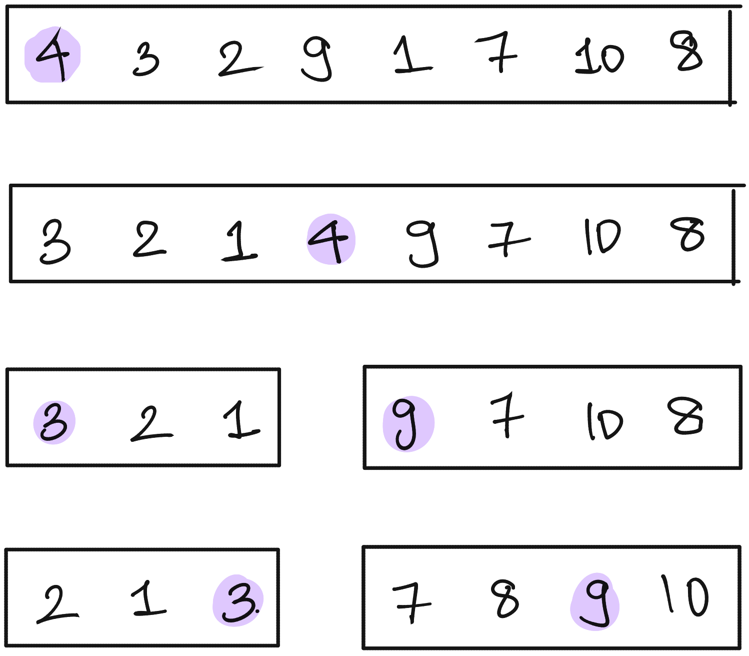


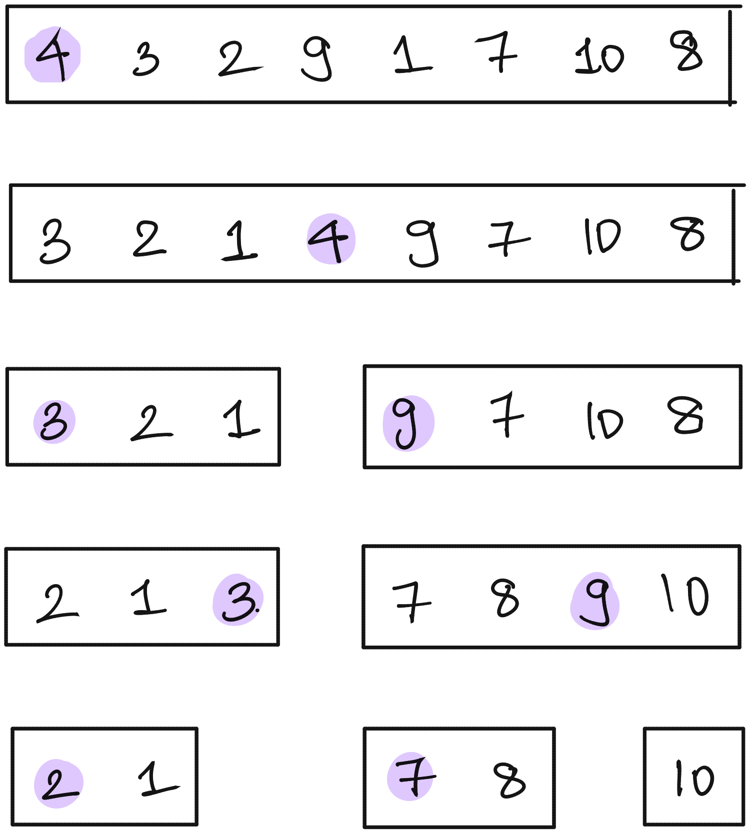
4	3	2	9	1	7	10	8

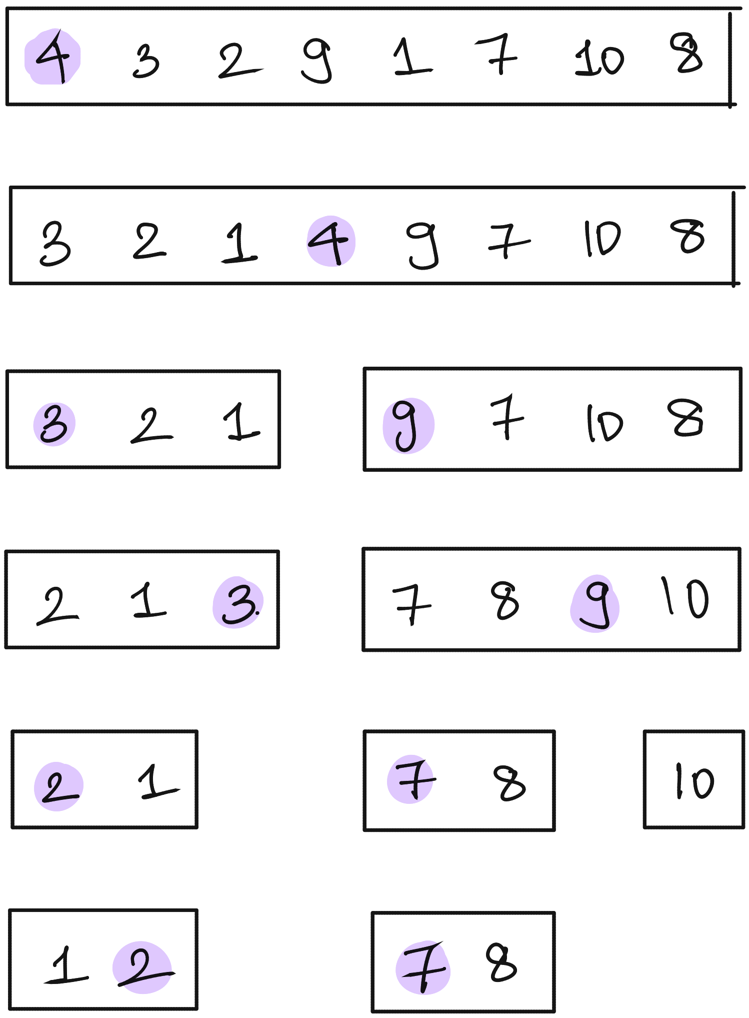
4	3	2	9	1	7	10	8

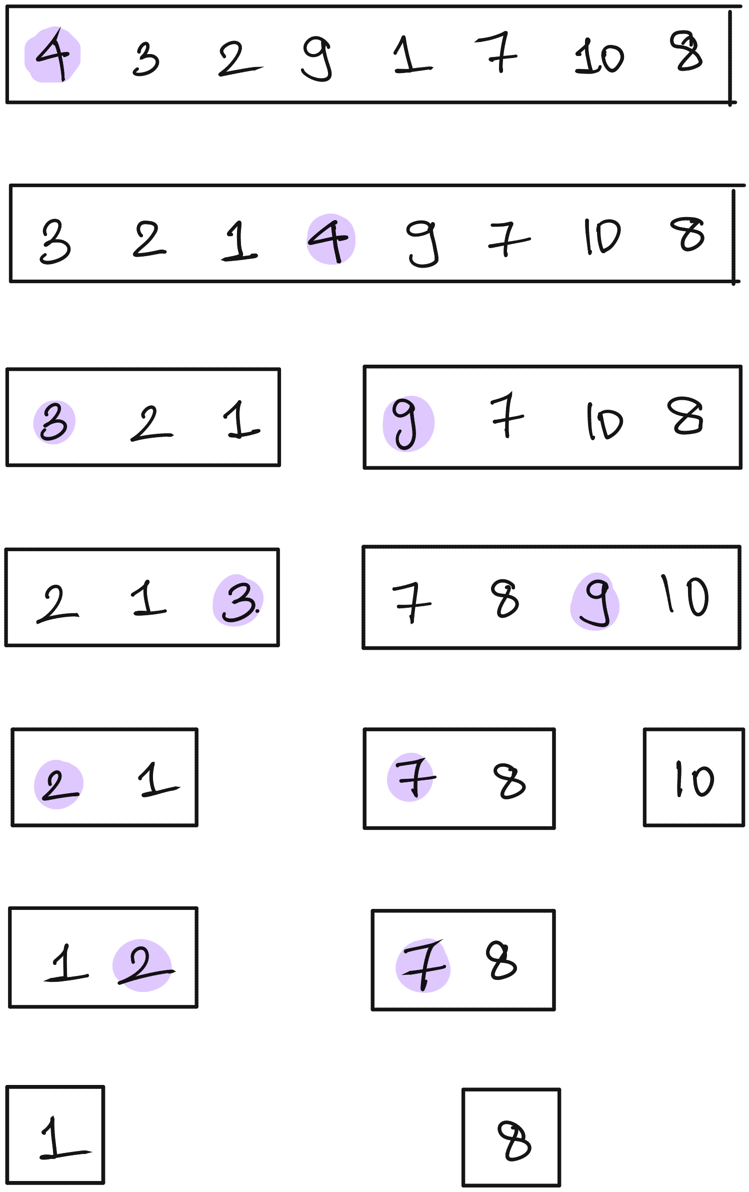
3	2	1	4	9	7	10	8

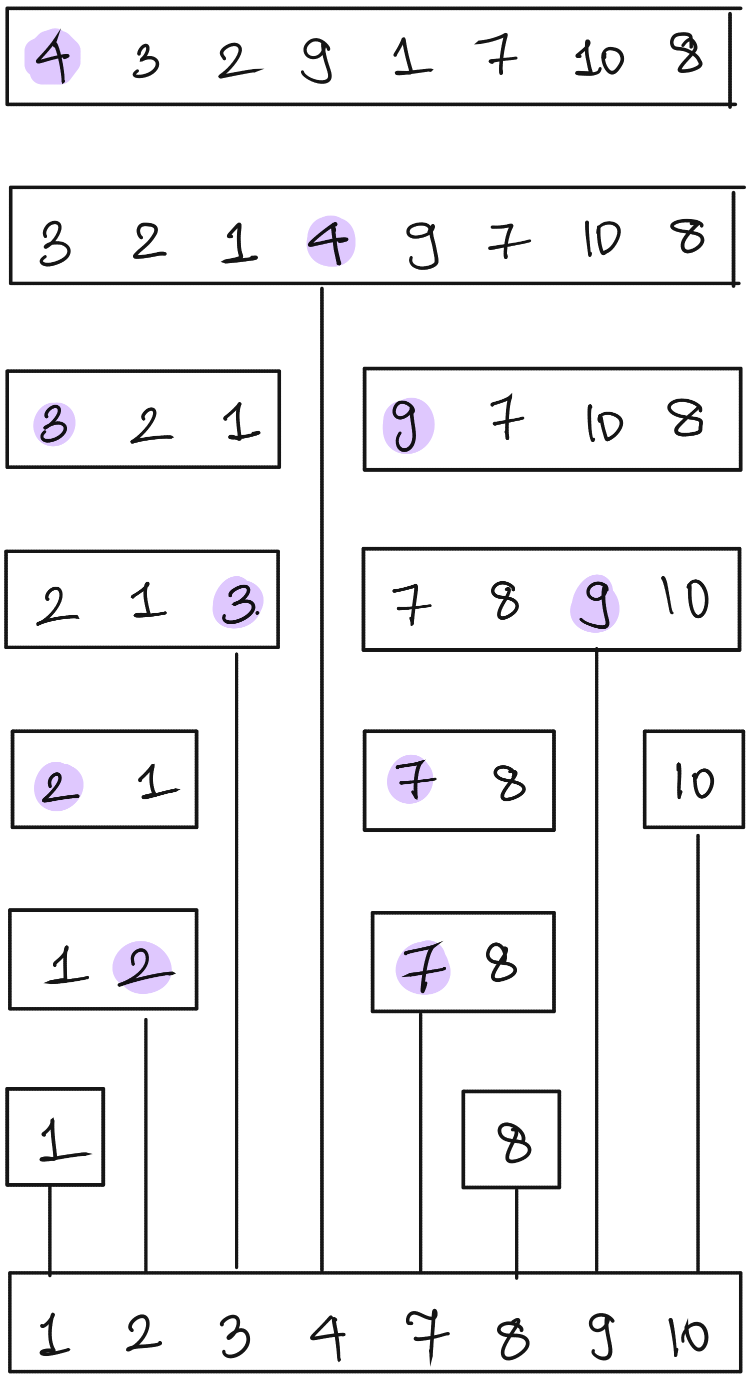

4 3 2 9 1 7 10 8
3 2 1 4 9 7 10 8
3 2 1
9 7 10 8


4 3 2 9 1 7 10 8
3 2 1 4 9 7 10 8
3 2 1
9 7 10 8
2 1 3
7 8 9 10


4 3 2 9 1 7 10 8
3 2 1 4 9 7 10 8
3 2 1
9 7 10 8
2 1 3
7 8 9 10
2 1
7 8
10


4 3 2 9 1 7 10 8
3 2 1 4 9 7 10 8
3 2 1
9 7 10 8
2 1 3
7 8 9 10
2 1
7 8
10
1 2
7 8


4 3 2 9 1 7 10 8
3 2 1 4 9 7 10 8
3 2 1
9 7 10 8
2 1 3
7 8 9 10
2 1
7 8
10
1 2
7 8
1
8


4 3 2 9 1 7 10 8
3 2 1 4 9 7 10 8
3 2 1
9 7 10 8
2 1 3
7 8 9 10
2 1
7 8
10
1 2
7 8
1
8
1 2 3 4 7 8 9 10

WORST CASE INPUT OF QUICKSORT

WORST CASE INPUT OF QUICKSORT

1 2 3 4 .... n

WORST CASE INPUT OF QUICKSORT

1 2 3 4 .... n

WORST CASE INPUT OF QUICKSORT

1 2 3 4 .... n

2 3 4 .... n

WORST CASE INPUT OF QUICKSORT

1 2 3 4 .... n

2 3 4 .... n

3 4 .... n

⋱

AND SO ON

WORST CASE INPUT OF QUICKSORT

1 2 3 4 .... n n

2 3 4 .... n $n-1$

3 4 .... n $n-2$

⋮ ⋮

AND SO ON

WORST CASE INPUT OF QUICKSORT

1 2 3 4 .... n	n
2 3 4 .... n	$n-1$
3 4 .... n	$n-2$
⋱	⋮

AND SO ON

TIME TAKEN = $n + n-1 + \dots + 1$

$= \frac{n(n+1)}{2}$

$= O(n^2)$.

BEST CASE INPUT FOR QUICKSORT.

BEST CASE INPUT FOR QUICKSORT.

INPUT FOR WHICH THE PIVOT LANDS UP IN THE MIDDLE OF THE ARRAY EVERYTIME.

BEST CASE INPUT FOR QUICKSORT.

INPUT FOR WHICH THE PIVOT LANDS UP IN THE MIDDLE OF THE ARRAY EVERYTIME.

THEN QUICKSORT WORKS LIKE MERGESORT.

BEST CASE INPUT FOR QUICKSORT.

INPUT FOR WHICH THE PIVOT LANDS UP IN THE MIDDLE OF THE ARRAY EVERYTIME.

THEN QUICKSORT WORKS LIKE MERGESORT.

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

BEST CASE INPUT FOR QUICKSORT.
INPUT FOR WHICH THE PIVOT LANDS UP IN THE MIDDLE OF THE ARRAY EVERYTIME.

THEN QUICKSORT WORKS LIKE MERGESORT.

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

$$T(n) = O(n \log n).$$

BEST CASE INPUT FOR QUICKSORT.

INPUT FOR WHICH THE PIVOT LANDS UP IN THE MIDDLE OF THE ARRAY EVERYTIME.

THEN QUICKSORT WORKS LIKE MERGESORT.

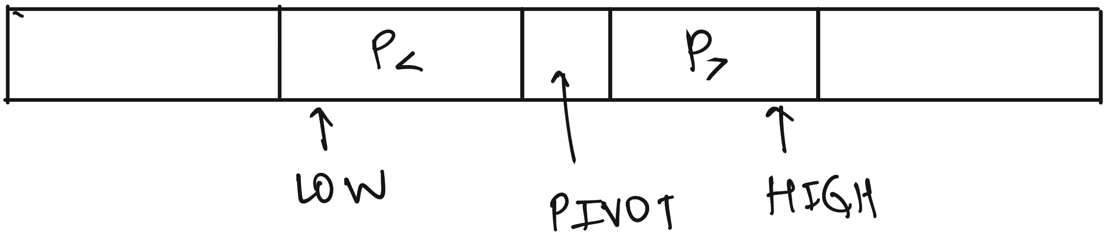
$$T(n) \leq T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + cn$$

$$T(n) = O(n\log n).$$

AVERAGE CASE: $O(n \log n)$

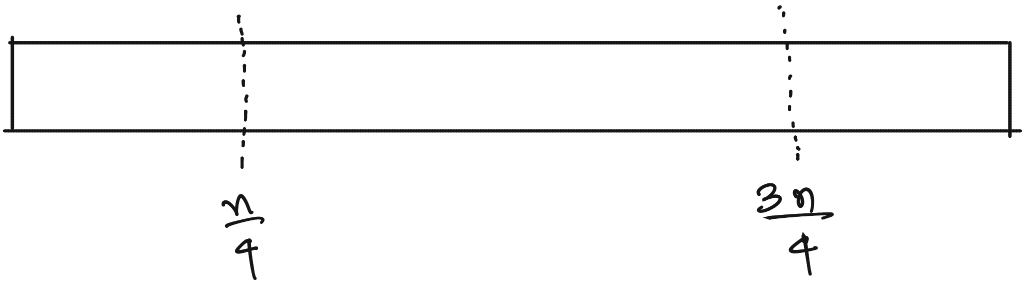
SEE NOTES.

```
QUICKSORT(A, LOW, HIGH)
{       IF (LOW = HIGH)
            RETURN;
        PIVOT ← RANDOM CELL IN A[LOW...HIGH]
        $P_<$ ← NUMBERS LESS THAN PIVOT IN
                    A[LOW..... HIGH]
        $P_>$ ← NUMBERS GREATER THAN PIVOT IN
                    A[LOW .... HIGH]
```



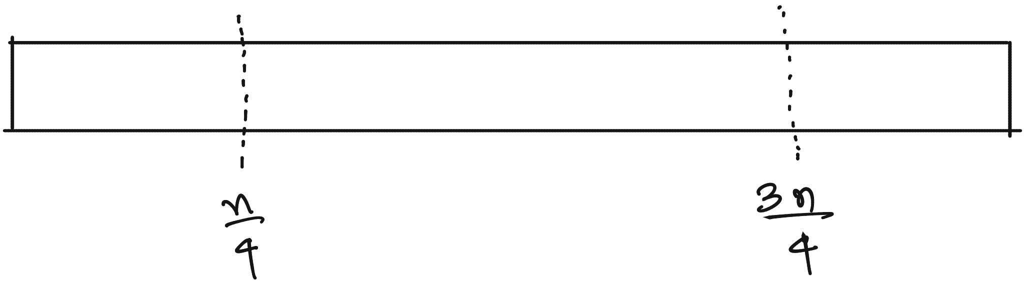
```
        QUICKSORT(A, LOW, LOW + $|P_<|$ - 1);
        QUICKSORT(A, LOW + $|P_<|$ + 1, HIGH);
}
```

RANDOMIZED ALGORITHM.



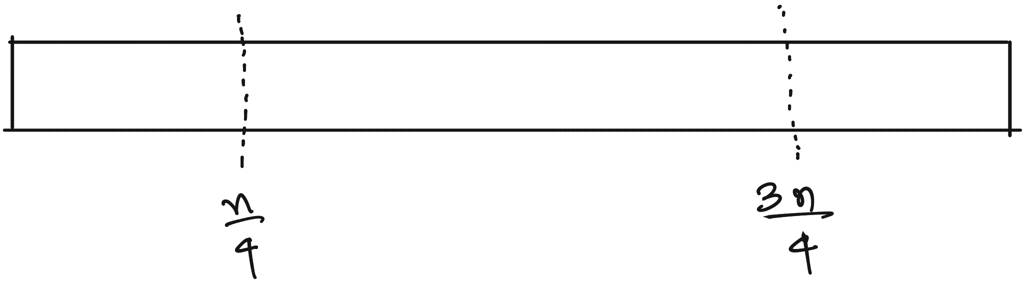
IF THE PIVOT K^{th} MINIMINUM $K \in \left[\frac{n}{4}, \frac{3n}{4}\right]$, THEN IT WILL

LAND UP SOMEWHERE IN THE MIDDLE



IF THE PIVOT K^{th} MINIMINUM $K \in \left[\frac{n}{4}, \frac{3n}{4}\right]$, THEN IT WILL LAND UP SOMEWHERE IN THE MIDDLE

PR[PIVOT LIES SOMEWHERE IN MIDDLE]

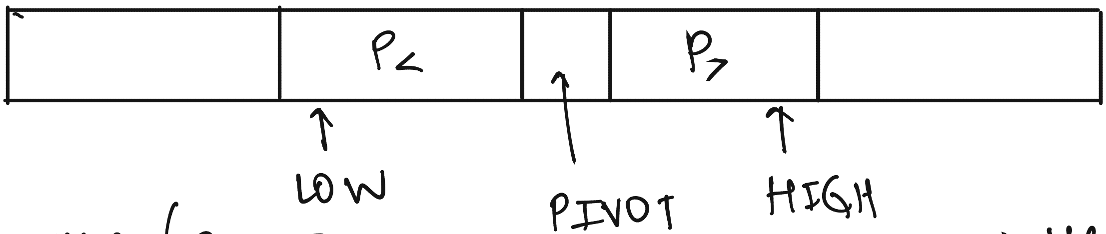


IF THE PIVOT K^{th} MINIMINUM $K \in \left[\frac{n}{4}, \frac{3n}{4}\right]$, THEN IT WILL LAND UP SOMEWHERE IN THE MIDDLE

PR[PIVOT LIES SOMEWHERE IN MIDDLE]

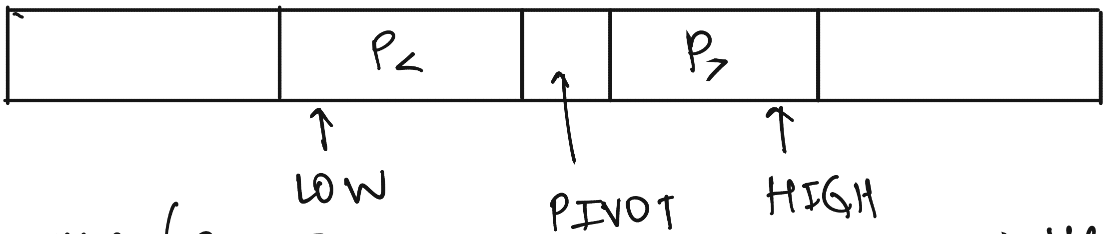
$$= \frac{1}{2}$$

```
QUICKSORT(A, LOW, HIGH)
{       IF (LOW = HIGH)
            RETURN;
  DO{
        PIVOT ← RANDOM CELL IN A[LOW...HIGH]
        $P_<$ ← NUMBERS LESS THAN PIVOT IN
                  A[LOW..... HIGH]
        $P_>$ ← NUMBERS GREATER THAN PIVOT IN
                  A[LOW .... HIGH]
```



```
  } WHILE (PIVOT DOES NOT LAND SOMEWHERE IN MIDDLE)
        QUICKSORT(A, LOW, LOW + $|P_<|$ - 1);
        QUICKSORT(A, LOW + $|P_<|$ + 1, HIGH);
}
```

```
QUICKSORT ( A , LOW, HIGH)
{       IF ( LOW = HIGH)
                RETURN;
  DO{
        PIVOT ← RANDOM CELL IN A[LOW...HIGH]
        $P_<$ ← NUMBERS LESS THAN PIVOT IN
                    A[LOW..... HIGH]
        $P_>$ ← NUMBERS GREATER THAN PIVOT IN
                    A[LOW .... HIGH]
```



```
  } WHILE (PIVOT DOES NOT LAND SOMEWHERE IN MIDDLE)
        QUICKSORT ( A , LOW, LOW + $|P_<|$ - 1);
        QUICKSORT ( A, LOW + $|P_<|$ + 1 , HIGH);
}
```

WE ARE DOING RANDOM TRIALS TO FIND AN APPROPRIATE PIVOT.

Q: HOW MANY TIMES DO WE HAVE TO DO THIS TRIAL IN EXPECTATION?

Q: HOW MANY TIMES DO WE HAVE TO DO THIS TRIAL IN EXPECTATION ?

A:

EXPECTED NUMBER OF TRIALS (S)

$$= \frac{1}{2}(1) + \left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\cdot 2 + \left(\frac{1}{2}\right)^2\left(\frac{1}{2}\right)\cdot 3$$

$$+ \left(\frac{1}{2}\right)^3\left(\frac{1}{2}\right)\cdot 4 + \cdots\cdots$$

Q: HOW MANY TIMES DO WE HAVE TO DO THIS TRIAL IN EXPECTATION?

A:

EXPECTED NUMBER OF TRIALS (s)

$$= \frac{1}{2}(1) + \left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\cdot 2 + \left(\frac{1}{2}\right)^2\left(\frac{1}{2}\right)\cdot 3$$

$$+ \left(\frac{1}{2}\right)^3\left(\frac{1}{2}\right)\cdot 4 + \ldots\ldots$$

$$S = \frac{1}{2} + \left(\frac{1}{2}\right)^2\cdot 2 + \left(\frac{1}{2}\right)^3\cdot 3 + \left(\frac{1}{2}\right)^4\cdot 4 + \ldots.$$

Q: HOW MANY TIMES DO WE HAVE TO DO THIS TRIAL IN EXPECTATION?

A:

EXPECTED NUMBER OF TRIALS (S)

$$= \frac{1}{2}(1) + \left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\cdot 2 + \left(\frac{1}{2}\right)^2\left(\frac{1}{2}\right)\cdot 3$$

$$+ \left(\frac{1}{2}\right)^3\left(\frac{1}{2}\right)\cdot 4 + \ldots\ldots$$

$$S = \frac{1}{2} + \left(\frac{1}{2}\right)^2\cdot 2 + \left(\frac{1}{2}\right)^3\cdot 3 + \left(\frac{1}{2}\right)^4\cdot 4 + \ldots.$$

$$S = 2$$

Q: HOW MANY TIMES DO WE HAVE TO DO THIS TRIAL IN EXPECTATION?

A:

EXPECTED NUMBER OF TRIALS (s)

$$= \frac{1}{2}(1) + \left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\cdot 2 + \left(\frac{1}{2}\right)^2\left(\frac{1}{2}\right)\cdot 3$$

$$+ \left(\frac{1}{2}\right)^3\left(\frac{1}{2}\right)\cdot 4 + \cdots\cdots$$

$$S = \frac{1}{2} + \left(\frac{1}{2}\right)^2\cdot 2 + \left(\frac{1}{2}\right)^3\cdot 3 + \left(\frac{1}{2}\right)^4\cdot 4 + \cdots\cdot$$

$$S = 2$$

EXPECTED RUNNING TIME FOR FINDING THE APPROPRIATE PIVOT $= 2cn$

Q: HOW MANY TIMES DO WE HAVE TO DO THIS TRIAL IN EXPECTATION?

A:

EXPECTED NUMBER OF TRIALS (s)

$$= \frac{1}{2}(1) + \left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\cdot 2 + \left(\frac{1}{2}\right)^2\left(\frac{1}{2}\right)\cdot 3$$

$$+ \left(\frac{1}{2}\right)^3\left(\frac{1}{2}\right)\cdot 4 + \cdots\cdots$$

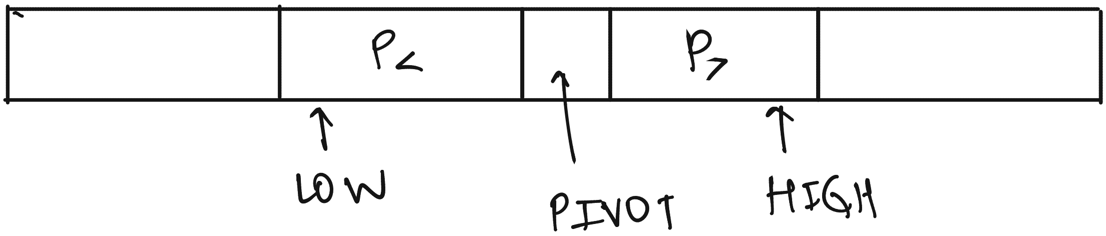
$$S = \frac{1}{2} + \left(\frac{1}{2}\right)^2\cdot 2 + \left(\frac{1}{2}\right)^3\cdot 3 + \left(\frac{1}{2}\right)^4\cdot 4 + \cdots\cdots$$

$$S = 2$$

EXPECTED RUNNING TIME FOR FINDING THE APPROPRIATE PIVOT $= 2cn$

$$T(n) \leq T\left(\frac{n}{4}\right) + T\left(\frac{3n}{4}\right) + 2cn$$

```
QUICKSORT ( A , LOW, HIGH)
{     IF ( LOW = HIGH)
          RETURN;
      PIVOT ← A[LOW]
      $P_<$ ← NUMBERS LESS THAN PIVOT IN
              A[LOW..... HIGH]
      $P_>$ ← NUMBERS GREATER THAN PIVOT IN
              A[LOW .... HIGH]
```

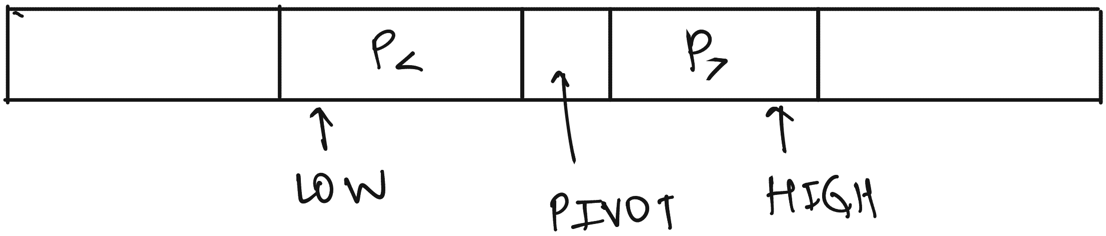


```
      QUICKSORT ( A , LOW, LOW + $|P_<|$ - 1);
      QUICKSORT ( A, LOW + $|P_<|$ + 1 , HIGH) ;
}
```

```
QUICKSORT(A, LOW, HIGH)
{       IF (LOW = HIGH)
            RETURN;
        PIVOT ← A[LOW]
        $P_<$ ← NUMBERS LESS THAN PIVOT IN
                    A[LOW..... HIGH]
        $P_>$ ← NUMBERS GREATER THAN PIVOT IN
                    A[LOW .... HIGH]
```

O(HIGH-LOW+1)

EXTRA MEMORY

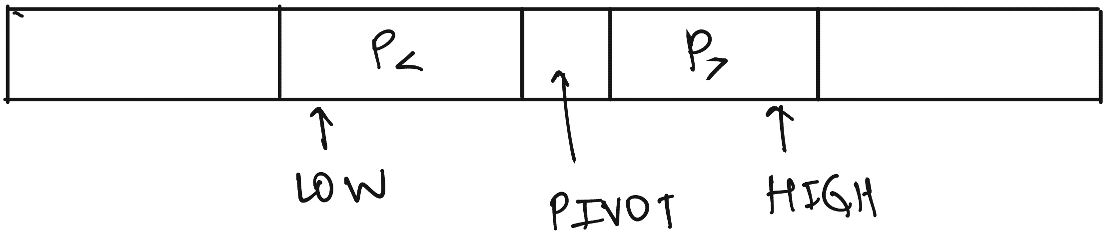


```
        QUICKSORT(A, LOW, LOW + $|P_<|$ - 1);
        QUICKSORT(A, LOW + $|P_<|$ + 1, HIGH);
}
```

A PROBLEM WITH THE ABOVE CODE IS THAT IT USES EXTRA MEMORY.

```
QUICKSORT(A, LOW, HIGH)
{       IF (LOW = HIGH)
            RETURN;
        PIVOT ← A[LOW]
        P< ← NUMBERS LESS THAN PIVOT IN
                 A[LOW..... HIGH]
        P> ← NUMBERS GREATER THAN PIVOT IN
                 A[LOW .... HIGH]
```

O(HIGH-LOW+1) EXTRA MEMORY



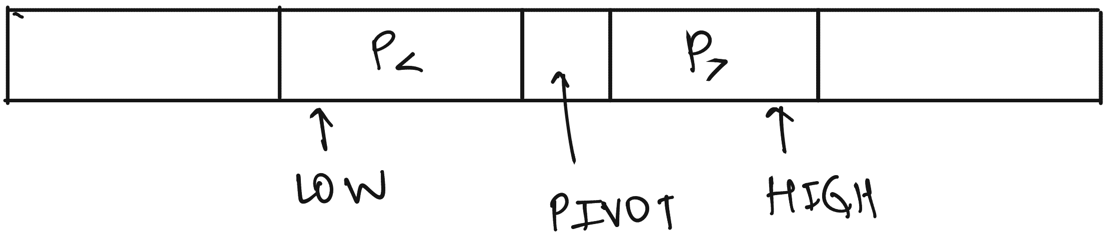
```
        QUICKSORT(A, LOW, LOW + |P<| - 1);
        QUICKSORT(A, LOW + |P<| + 1, HIGH);
}
```

A PROBLEM WITH THE ABOVE CODE IS THAT IT USES EXTRA MEMORY.

DONT WANT TO USE HUGE EXTRA MEMORY?

```
QUICKSORT(A, LOW, HIGH)
{       IF (LOW = HIGH)
            RETURN;
        PIVOT ← A[LOW]
        $P_<$ ← NUMBERS LESS THAN PIVOT IN
                A[LOW..... HIGH]
        $P_>$ ← NUMBERS GREATER THAN PIVOT IN
                A[LOW.... HIGH]
```

$O(HIGH - LOW + 1)$
EXTRA MEMORY



```
        QUICKSORT(A, LOW, LOW + $|P_<|$ - 1);
        QUICKSORT(A, LOW + $|P_<|$ + 1, HIGH);
}
```

A PROBLEM WITH THE ABOVE CODE IS THAT IT USES EXTRA MEMORY.

DONT WANT TO USE HUGE EXTRA MEMORY?

CAN WE PARTITION THE ARRAY USING O(1) EXTRA MEMORY ?

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING $O(1)$ EXTRA MEMORY IN $O(n)$ TIME.

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING $O(1)$ EXTRA MEMORY IN $O(n)$ TIME.

1) FIND THE FINAL POSITION OF PIVOT IN ONE PASS

PIVOT → 4 $\underset{\overset{\uparrow}{i}}{1}$ 5 2 6 7 3 9 $c = 0$

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING $O(1)$ EXTRA MEMORY IN $O(n)$ TIME.

1) FIND THE FINAL POSITION OF PIVOT IN ONE PASS

PIVOT → 4 1 5 2 6 7 3 9 $c = 1$

↑ i (under 5)

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING $O(1)$ EXTRA MEMORY IN $O(n)$ TIME.

1) FIND THE FINAL POSITION OF PIVOT IN ONE PASS

PIVOT → 4 1 5 2 6 7 3 9 $c = 1$

↑

i

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING O(1) EXTRA MEMORY IN O(n) TIME.

1) FIND THE FINAL POSITION OF PIVOT IN ONE PASS

$$\text{PIVOT} \longrightarrow 4 \quad 1 \quad 5 \quad 2 \quad \underset{\overset{\uparrow}{i}}{6} \quad 7 \quad 3 \quad 9 \qquad\qquad c = 2$$

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING $O(1)$ EXTRA MEMORY IN $O(n)$ TIME.

1) FIND THE FINAL POSITION OF PIVOT IN ONE PASS

PIVOT → 4 1 5 2 6 7 3 9 $\qquad$ $c = 3$

$\uparrow$

i

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING $O(1)$ EXTRA MEMORY IN $O(n)$ TIME.

1) FIND THE FINAL POSITION OF PIVOT IN ONE PASS

PIVOT → 4 1 5 2 6 7 3 9 $\quad\quad$ $c = 3$

$\uparrow$

i

```
c ← 0
PIVOT ← A[1]
FOREACH  i ← 2 to n
{   IF A[i] < PIVOT
    {
        c ← c+1
    }
}
```

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING $O(1)$ EXTRA MEMORY IN $O(n)$ TIME.

1) FIND THE FINAL POSITION OF PIVOT IN ONE PASS

PIVOT ⟶ 2 1 5 4 6 7 3 9 ↑ i $c = 3$

```
C ← 0
PIVOT ← A[1]
FOREACH i ← 2 to n
{
    IF A[i] < PIVOT
    {
        C ← C+1
    }
}
SWAP(A[1], A[c+1])
```

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING $O(1)$ EXTRA MEMORY IN $O(n)$ TIME.

1) FIND THE FINAL POSITION OF PIVOT IN ONE PASS

PIVOT → 2 1 5 4 6 7 3 9 $\quad c = 3$

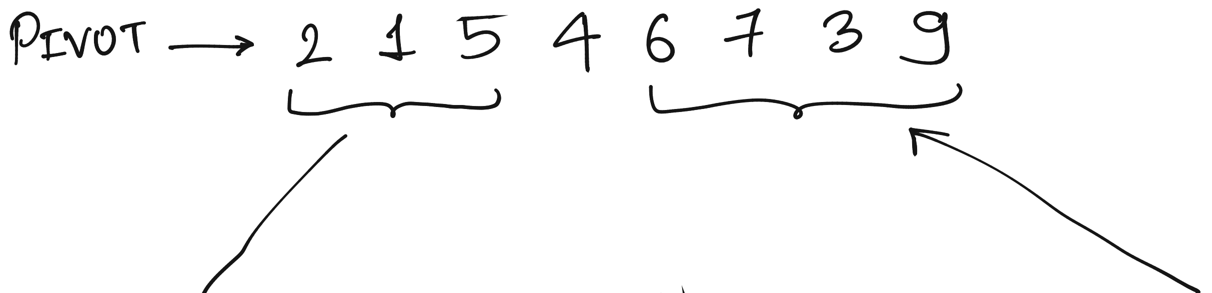
$\uparrow i$

```
C ← 0
PIVOT ← A[1]
FOREACH i ← 2 to n
{
    IF A[i] < PIVOT
    {
        C ← C+1
    }
}
SWAP(A[1], A[C+1])
```

THE PIVOT IS NOW AT ITS FINAL PLACE. NOW WE HAVE PUT ALL NUMBERS LESS THAN 4 TO THE LEFT OF 4 & ALL NUMBERS GREATER THAN 4 TO THE RIGHT OF 4.

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING $O(1)$ EXTRA MEMORY IN $O(n)$ TIME.

1) FIND THE FINAL POSITION OF PIVOT IN ONE PASS

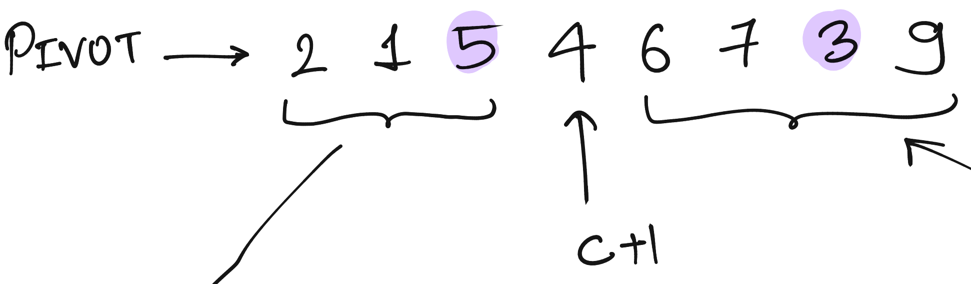


IF THERE IS ANY NUMBER > 4 HERE, PUT IT TO THE RIGHT OF 4

IF THERE IS ANY NUMBER < 4 HERE, PUT IT TO THE LEFT OF 4

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING O(1) EXTRA MEMORY IN O(n) TIME.

1) FIND THE FINAL POSITION OF PIVOT IN ONE PASS



IF THERE IS ANY NUMBER > 4 HERE, PUT IT TO THE RIGHT OF 4

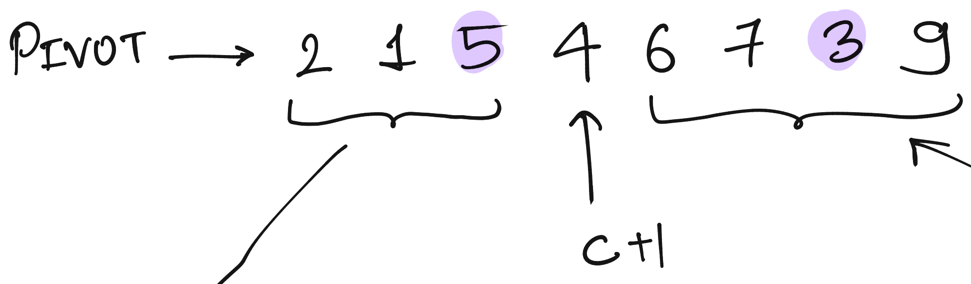
IF THERE IS ANY NUMBER < 4 HERE, PUT IT TO THE LEFT OF 4

$\#$ NUMBERS GREATER THAN PIVOT IN A[1....c] = 1

$\#$ NUMBERS LESS THAN PIVOT IN A[c+2...n] = 1

PROBLEM: IN QUICKSORT, PARTITION THE ARRAY USING O(1) EXTRA MEMORY IN O(n) TIME.

1) FIND THE FINAL POSITION OF PIVOT IN ONE PASS



IF THERE IS ANY NUMBER > 4 HERE, PUT IT TO THE RIGHT OF 4

IF THERE IS ANY NUMBER < 4 HERE, PUT IT TO THE LEFT OF 4

\# NUMBERS GREATER THAN PIVOT IN $A[1 \ldots c] = 1$

\# NUMBERS LESS THAN PIVOT IN $A[c+2 \ldots n] = 1$

IS # NUMBERS GREATER THAN PIVOT IN $A[1 \ldots c]$ = # NUMBERS LESS THAN PIVOT IN $A[c+2 \ldots n]$?

NUMBERS GREATER THAN PIVOT IN $A[1 \dots c]$
$= x$

NUMBERS LESS THAN PIVOT IN $A[c+2 \dots b]$
$= y$

ASSUME FOR CONTRADICTION THAT $x \neq y$.

NUMBERS GREATER THAN PIVOT IN A[1...c]

= x

NUMBERS LESS THAN PIVOT IN A[C+2...n]

= y

ASSUME FOR CONTRADICTION THAT $x \neq y$.

NUMBERS LESS THAN PIVOT IN A[1...n]

= # NUMBERS LESS THAN PIVOT IN A[1...c]

\+ # NUMBERS LESS THAN PIVOT IN A[C+2... n]

=

NUMBERS GREATER THAN PIVOT IN $A[1 \ldots c]$

$= x$

NUMBERS LESS THAN PIVOT IN $A[c+2 \ldots n]$

$= y$

ASSUME FOR CONTRADICTION THAT $x \neq y$.

NUMBERS LESS THAN PIVOT IN $A[1 \ldots n]$

= # NUMBERS LESS THAN PIVOT IN $A[1 \ldots c]$

\+ # NUMBERS LESS THAN PIVOT IN $A[c+2 \ldots n]$

$= c - x + y$

NUMBERS GREATER THAN PIVOT IN $A[1 \ldots c]$

$= x$

NUMBERS LESS THAN PIVOT IN $A[c+2 \ldots n]$

$= y$

ASSUME FOR CONTRADICTION THAT $x \neq y$.

NUMBERS LESS THAN PIVOT IN $A[1 \ldots n]$

$=$ # NUMBERS LESS THAN PIVOT IN $A[1 \cdots c]$

$+$ # NUMBERS LESS THAN PIVOT IN $A[c+2 \ldots n]$

$= c - x + y$

$\neq c$ (SINCE $x \neq y$)

NUMBERS GREATER THAN PIVOT IN A[1...C]

= x

NUMBERS LESS THAN PIVOT IN A[C+2..n]

= y

ASSUME FOR CONTRADICTION THAT $x \neq y$.

NUMBERS LESS THAN PIVOT IN A[1...n]

= # NUMBERS LESS THAN PIVOT IN A[1...C]

+ # NUMBERS LESS THAN PIVOT IN A[C+2...n]

$= C - x + y$

$\neq C$ (SINCE $x \neq y$)

BUT AT THE START WE HAD CALCULATED THAT # NUMBERS < PIVOT = C

NUMBERS GREATER THAN PIVOT IN $A[1 \ldots c]$

$= x$

NUMBERS LESS THAN PIVOT IN $A[c+2 \ldots n]$

$= y$

X ASSUMPTION MUST BE INCORRECT

ASSUME FOR CONTRADICTION THAT $x \neq y$.

NUMBERS LESS THAN PIVOT IN $A[1 \ldots n]$

= # NUMBERS LESS THAN PIVOT IN $A[1 \ldots c]$

+ # NUMBERS LESS THAN PIVOT IN $A[c+2 \ldots n]$

$= c - x + y$

$\neq c$ (SINCE $x \neq y$)

BUT AT THE START WE HAD CALCULATED THAT # NUMBERS $<$ PIVOT $= c$

NUMBERS GREATER THAN PIVOT IN $A[1 \ldots c]$

$= x$

NUMBERS LESS THAN PIVOT IN $A[c+2 \ldots n]$

$= y$

X ASSUMPTION MUST BE INCORRECT

ASSUME FOR CONTRADICTION THAT $x \neq y$.

NUMBERS LESS THAN PIVOT IN $A[1 \ldots n]$

= # NUMBERS LESS THAN PIVOT IN $A[1 \cdots c]$

\+ # NUMBERS LESS THAN PIVOT IN $A[c+2 \ldots n]$

$= c - x + y$

$\neq c$ (SINCE $x \neq y$)

BUT AT THE START WE HAD CALCULATED THAT # NUMBERS < PIVOT = C

LEMMA : # NUMBERS GREATER THAN PIVOT IN $A[1 \ldots c]$ = # NUMBERS LESS THAN PIVOT IN $A[c+2 \ldots n]$.

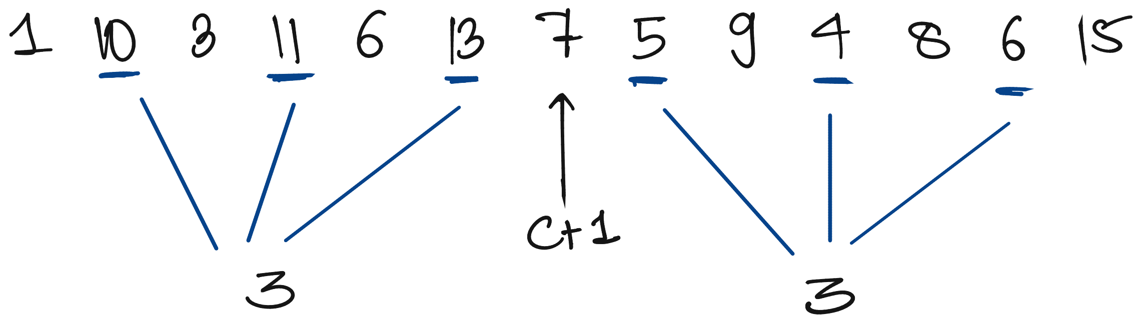
NOW WE WILL USE THIS NICE OBSERVATION

1 10 3 11 6 13 7 5 9 4 8 6 15

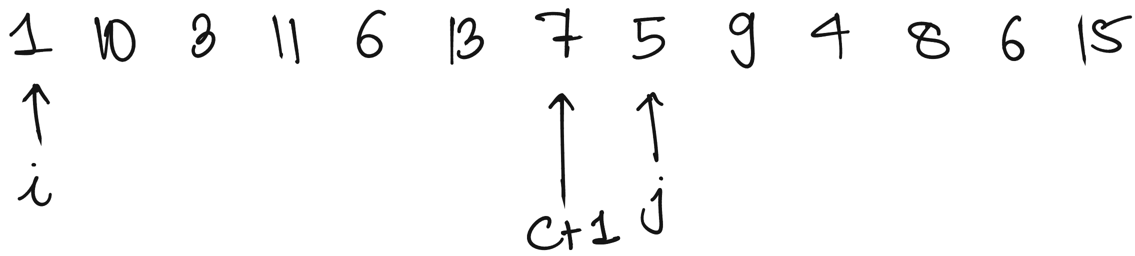
↑

C+1

NOW WE WILL USE THIS NICE OBSERVATION



NOW WE WILL USE THIS NICE OBSERVATION

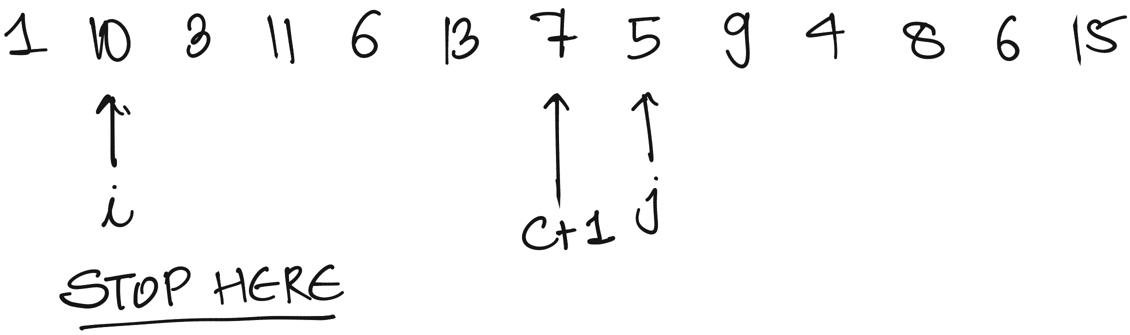


COUNTER i MOVES FROM $1 \dots c$

COUNTER j MOVES FROM $c+2 \dots n$

USING COUNTER i WE FIND A NUMBER WHICH IS GREATER THAN PIVOT IN $A[1 \dots c]$

NOW WE WILL USE THIS NICE OBSERVATION



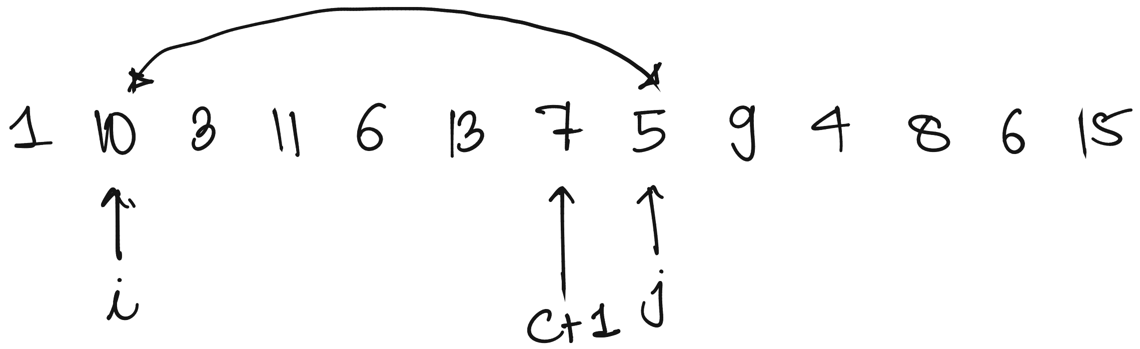
COUNTER i MOVES FROM $1 \ldots c$

COUNTER j MOVES FROM $c+2 \ldots n$

USING COUNTER i WE FIND A NUMBER WHICH IS GREATER THAN PIVOT IN $A[1 \ldots c]$

USING COUNTER j WE FIND A NUMBER WHICH IS LESS THAN PIVOT

NOW WE WILL USE THIS NICE OBSERVATION



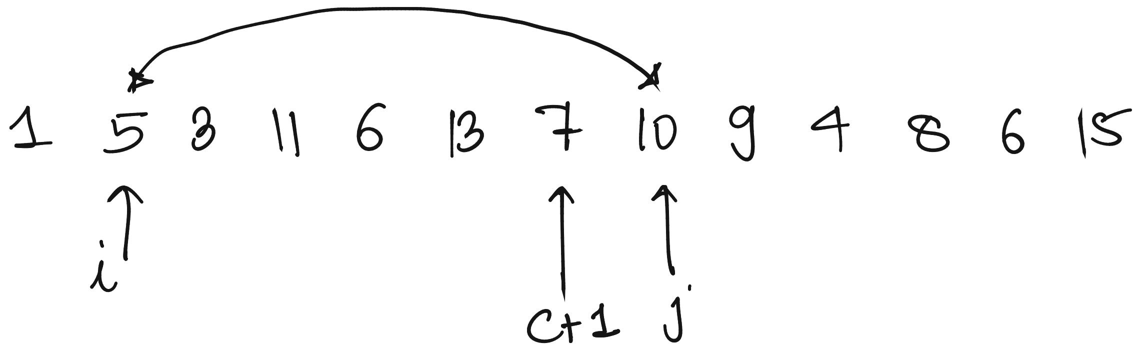
COUNTER i MOVES FROM $1 \ldots c$

COUNTER j MOVES FROM $c+2 \ldots n$

USING COUNTER i WE FIND A NUMBER WHICH IS GREATER THAN PIVOT IN $A[1 \ldots c]$

USING COUNTER j WE FIND A NUMBER WHICH IS LESS THAN PIVOT

NOW WE WILL USE THIS NICE OBSERVATION



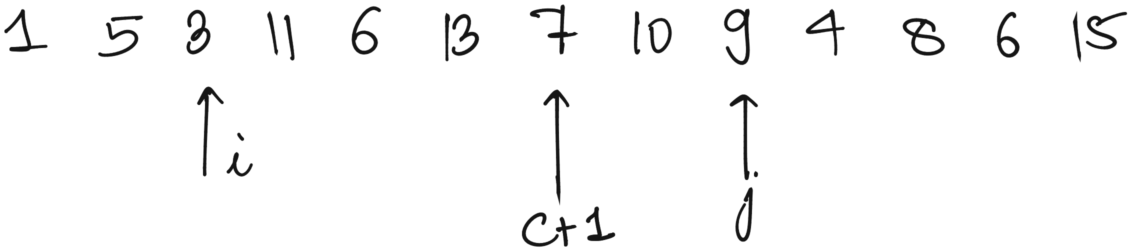
COUNTER i MOVES FROM $1 \ldots c$

COUNTER j MOVES FROM $c+2 \ldots n$

USING COUNTER i WE FIND A NUMBER WHICH IS GREATER THAN PIVOT IN $A[1 \ldots c]$

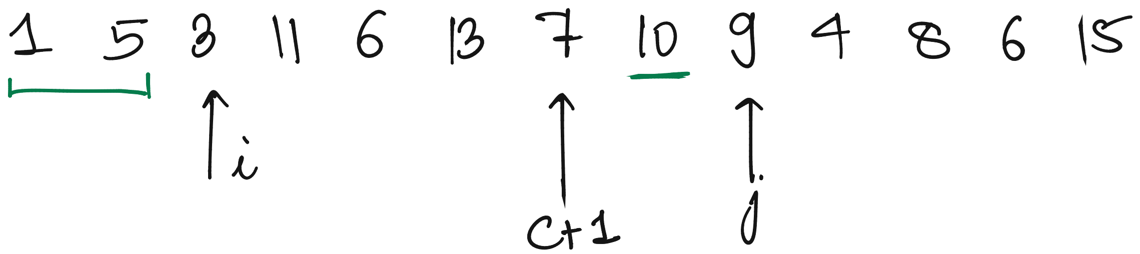
USING COUNTER j WE FIND A NUMBER WHICH IS LESS THAN PIVOT

NOW WE WILL USE THIS NICE OBSERVATION



FEW OBSERVATIONS:

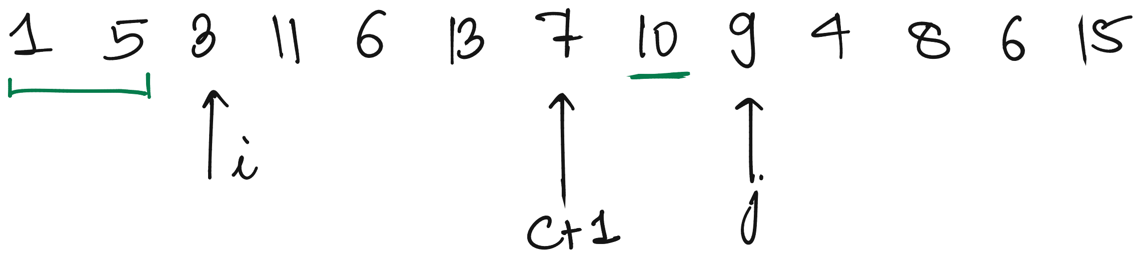
NOW WE WILL USE THIS NICE OBSERVATION



FEW OBSERVATIONS:

1) NUMBERS IN A[1.... i-1] ARE LESS THAN PIVOT
2) NUMBERS IN A[c+2... j-1] ARE GREATER THAN PIVOT.

NOW WE WILL USE THIS NICE OBSERVATION



FEW OBSERVATIONS:

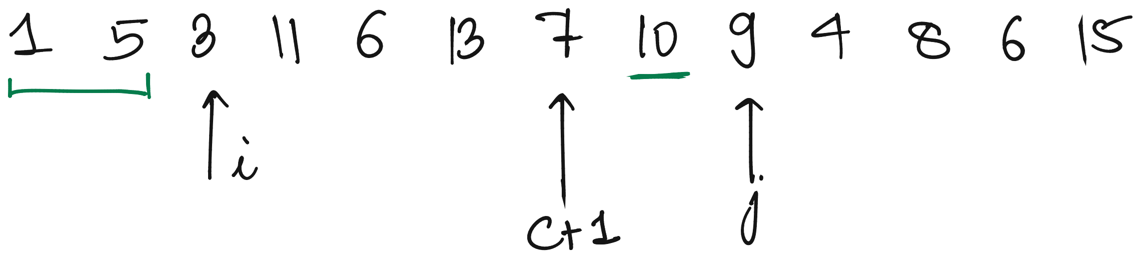
1) NUMBERS IN $A[1 \dots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[c+2 \dots j-1]$ ARE GREATER THAN PIVOT.

(3) # NUMBERS GREATER THAN PIVOT IN $A[i \dots c]$

NUMBERS LESS THAN PIVOT IN $A[j \dots n]$

NOW WE WILL USE THIS NICE OBSERVATION



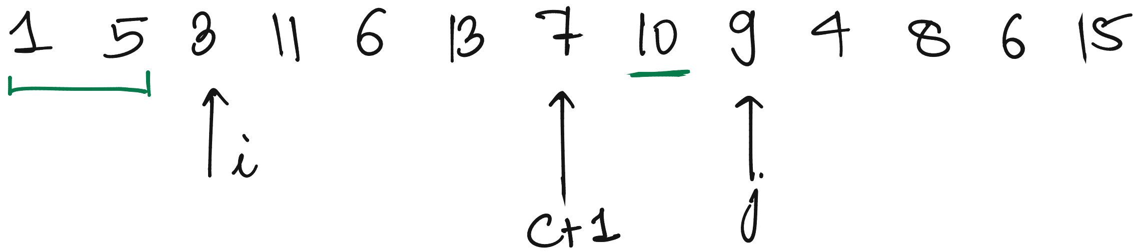
FEW OBSERVATIONS:

1) NUMBERS IN $A[1 \ldots i-1]$ ARE LESS THAN PIVOT

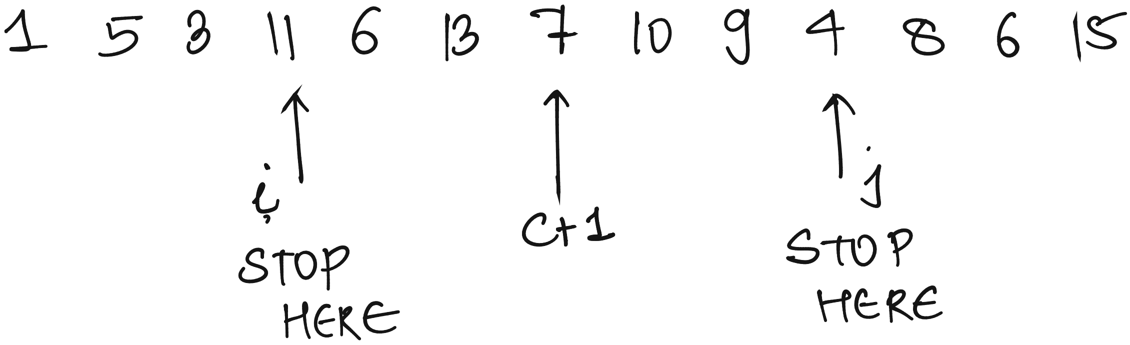
2) NUMBERS IN $A[C+2 \ldots j-1]$ ARE GREATER THAN PIVOT.

(3) # NUMBERS GREATER THAN PIVOT IN $A[i \ldots C]$ $=$ # NUMBERS LESS THAN PIVOT IN $A[j \ldots n]$

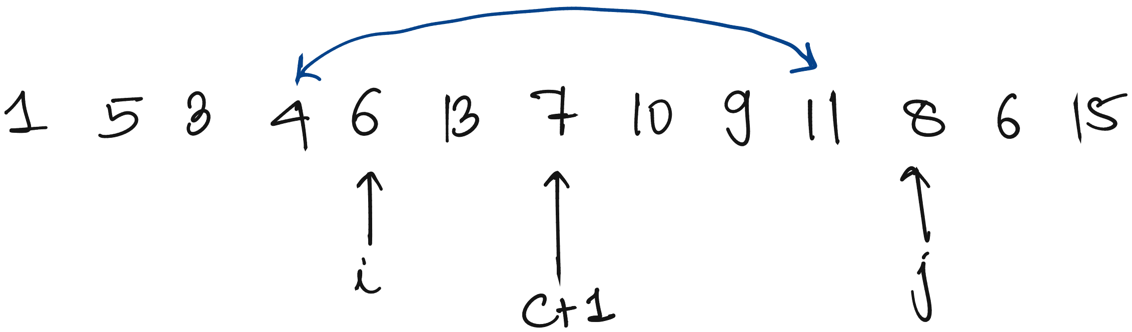
NOW WE WILL USE THIS NICE OBSERVATION



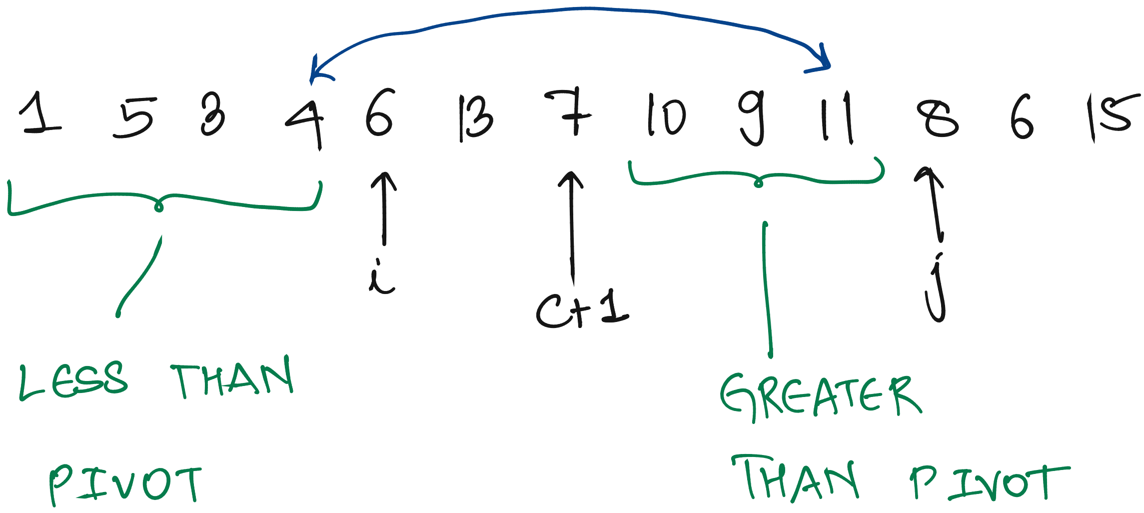
NOW WE WILL USE THIS NICE OBSERVATION



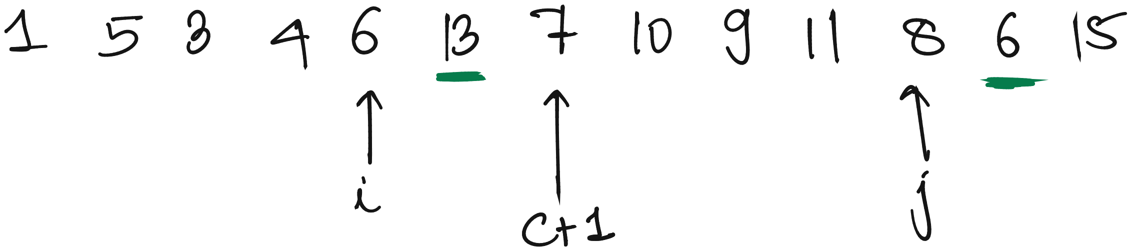
NOW WE WILL USE THIS NICE OBSERVATION



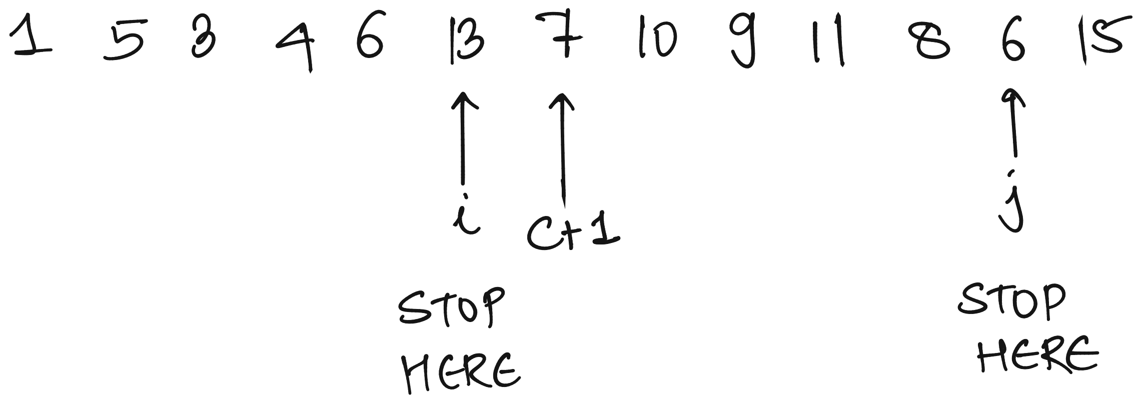
NOW WE WILL USE THIS NICE OBSERVATION



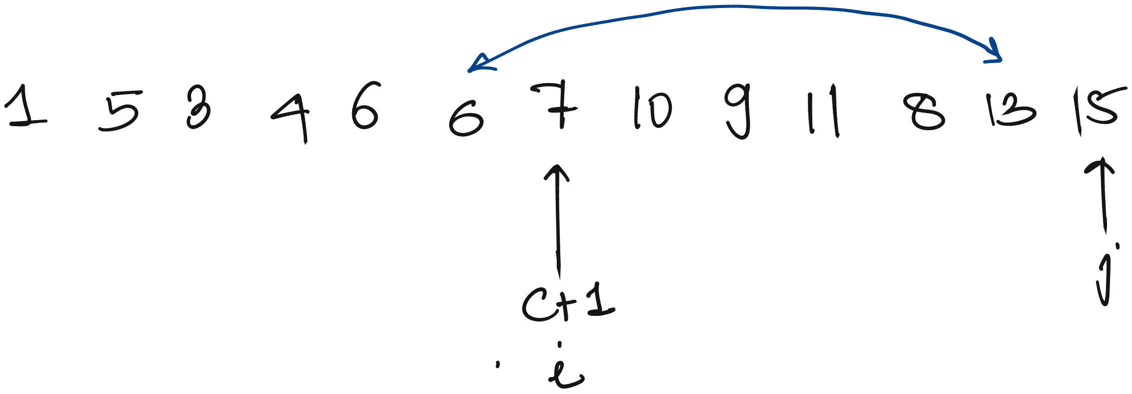
NOW WE WILL USE THIS NICE OBSERVATION



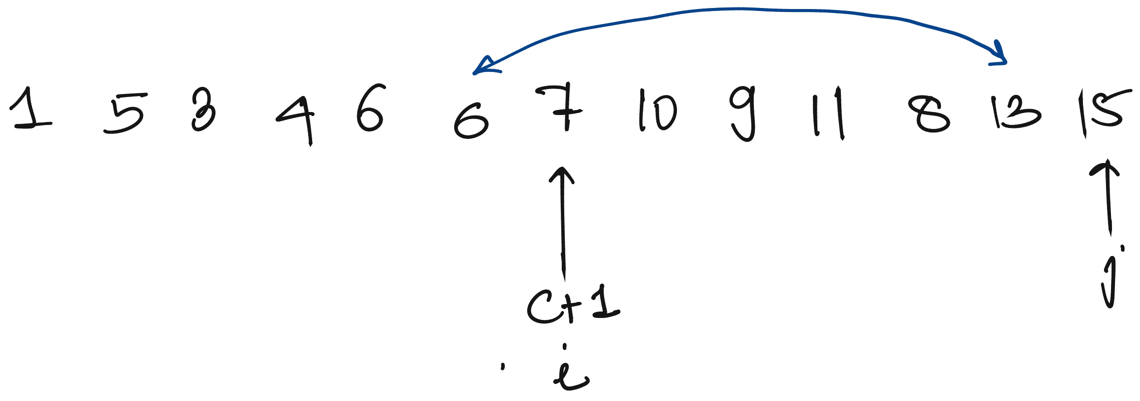
NOW WE WILL USE THIS NICE OBSERVATION



NOW WE WILL USE THIS NICE OBSERVATION



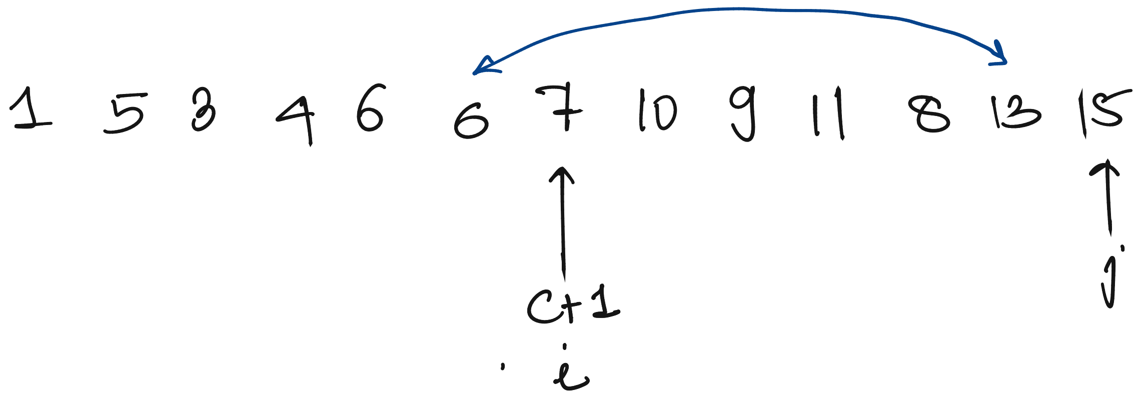
NOW WE WILL USE THIS NICE OBSERVATION



ASSUME THAT THE PIVOT IS AT INDEX c+1

```
i ← 1 ; j = c+2;
WHILE (TRUE)
{     WHILE ( A[i] < PIVOT AND i ≤ c)
            i ← i+1 ;
      WHILE ( A[j] > PIVOT AND j ≤ n)
            j ← j+1 ;
      IF ( i = c+1 OR j = n+1)
```

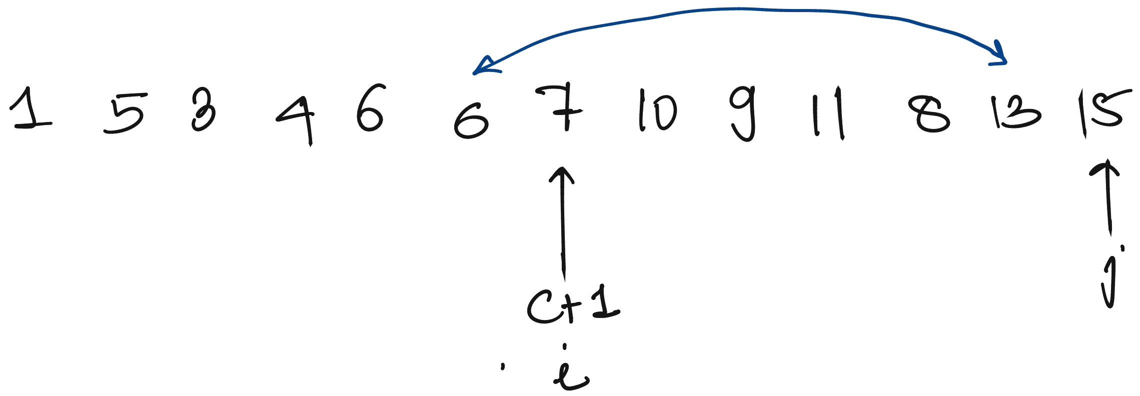
NOW WE WILL USE THIS NICE OBSERVATION



ASSUME THAT THE PIVOT IS AT INDEX C+1

```
i ← 1 ; j = c+2;
WHILE (TRUE)
{       WHILE ( A[i] < PIVOT AND i ≤ c)
                i ← i+1 ;
        WHILE ( A[j] > PIVOT AND j ≤ n)
                j ← j+1;
        IF ( i = C+1 OR j = n+1)
                BREAK;
        ELSE
        {       SWAP( A[i], A[j]) ;
                i ← i+1 ;
                j ← j+1 ;
        }
}
```

NOW WE WILL USE THIS NICE OBSERVATION



ASSUME THAT THE PIVOT IS AT INDEX c+1

```
i ← 1 ; j = c+2;
WHILE (TRUE)
{   WHILE ( A[i] < PIVOT AND i ≤ c)
        i ← i+1 ;
    WHILE ( A[j] > PIVOT AND j ≤ n)
        j ← j+1 ;
    IF ( i = c+1 OR j = n+1)
        BREAK ;
    ELSE
    {   SWAP( A[i], A[j]) ;
        i ← i+1 ;
        j ← j+1 ;
    }
}
```

RUNNING TIME = $O(n)$

OUR ALGORITM USES O(1) EXTRA MEMORY AND MAKES 2 PASSES OVER THE ARRAY

① PASS 1 : PLACE THE PIVOT AT ITS CORRECT POSITION

② PASS 2 : PARTITION THE OTHER NUMBERS IN THE ARRAY.

Q: CAN WE PARTITION THE ARRAY IN ONE PASS ?

OUR ALGORITM USES O(1) EXTRA MEMORY AND MAKES 2 PASSES OVER THE ARRAY

① **PASS 1** : PLACE THE PIVOT AT ITS CORRECT POSITION

② **PASS 2** : PARTITION THE OTHER NUMBERS IN THE ARRAY.

Q: CAN WE PARTITION THE ARRAY IN ONE PASS ?

Q: CAN WE DO AWAY WITH PASS 1.

OUR ALGORITM USES O(1) EXTRA MEMORY AND MAKES 2 PASSES OVER THE ARRAY

① PASS 1 : PLACE THE PIVOT AT ITS CORRECT POSITION

② PASS 2 : PARTITION THE OTHER NUMBERS IN THE ARRAY.

Q: CAN WE PARTITION THE ARRAY IN ONE PASS ?

Q: CAN WE DO AWAY WITH PASS 1.

Q: THEN HOW WILL WE FIND C.

PIVOT → 7 10 3 11 6 13 1 5 9 4 8 6 15

PIVOT → 7 10 3 11 6 13 1 5 9 4 8 6 15

↑

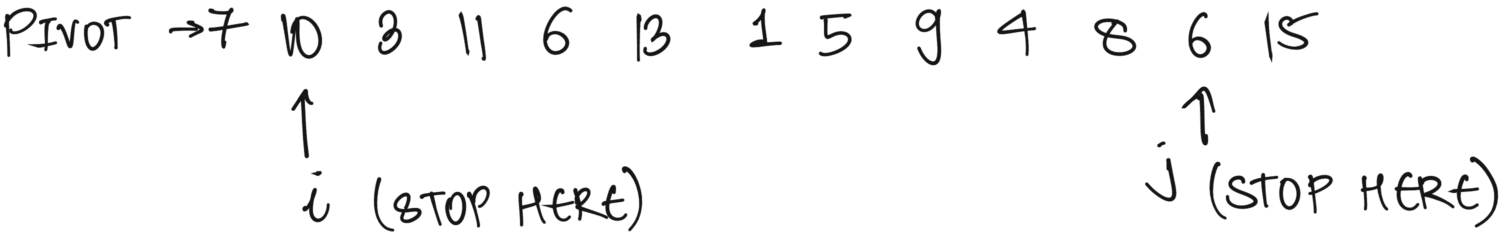
i

PIVOT →7 10 3 11 6 13 1 5 9 4 8 6 15

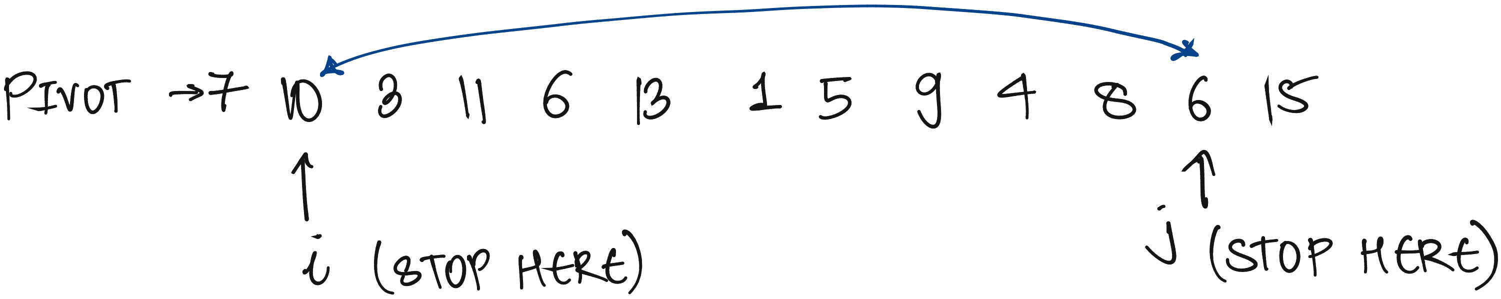
PIVOT →	7	10	3	11	6	13	1	5	9	4	8	6	15
		↑											↑
		i											*j*

① INCREMENT *i* TILL YOU FIND A NUMBER GREATER THAN PIVOT

② DECREMENT *j* TILL YOU FIND A NUMBER LESS THAN PIVOT



1. INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT
2. DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT



① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT

PIVOT → 7 6 3 11 6 13 1 5 9 4 8 10 15

↑ i (STOP HERE)

↑ j (STOP HERE)

① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT

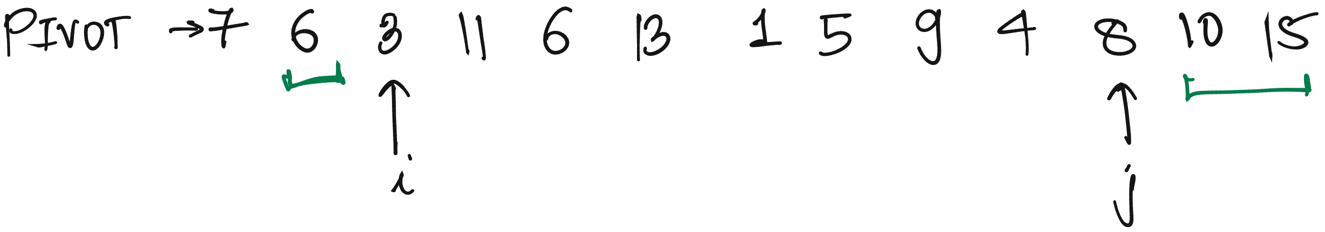
PIVOT → 7 6 3 11 6 13 1 5 9 4 8 10 15

↑ (under 3) i ↑ (under 8) j

(1) INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT

(2) DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT

FEW OBSERVATIONS :



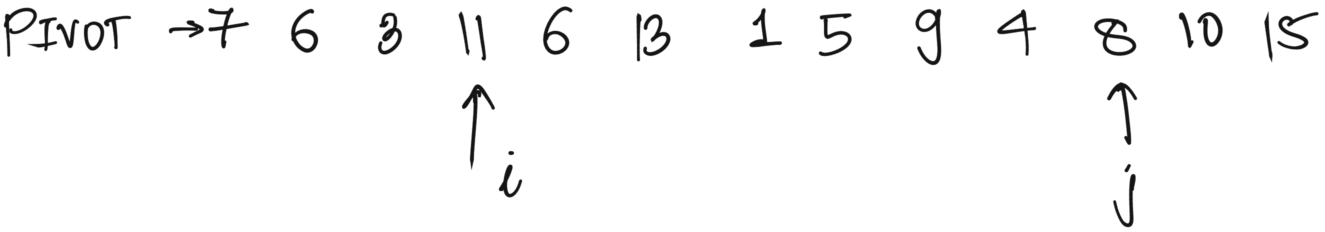
① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT

FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \ldots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j+1 \ldots n]$ ARE GREATER THAN PIVOT.



① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT

FEW OBSERVATIONS:

1) NUMBERS IN $A[2 \ldots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j+1 \ldots n]$ ARE GREATER THAN PIVOT.

PIVOT → 7 6 3 11 6 13 1 5 9 4 8 10 15

↑ i ↑ j

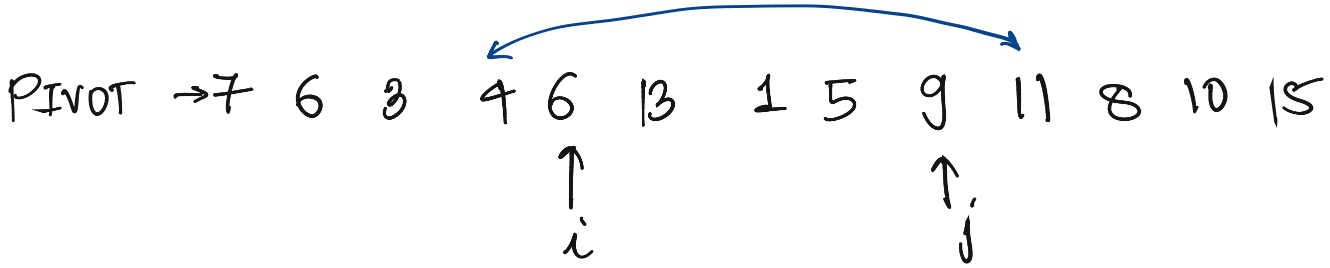
(1) INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT

(2) DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT

FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \ldots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j+1 \ldots n]$ ARE GREATER THAN PIVOT.



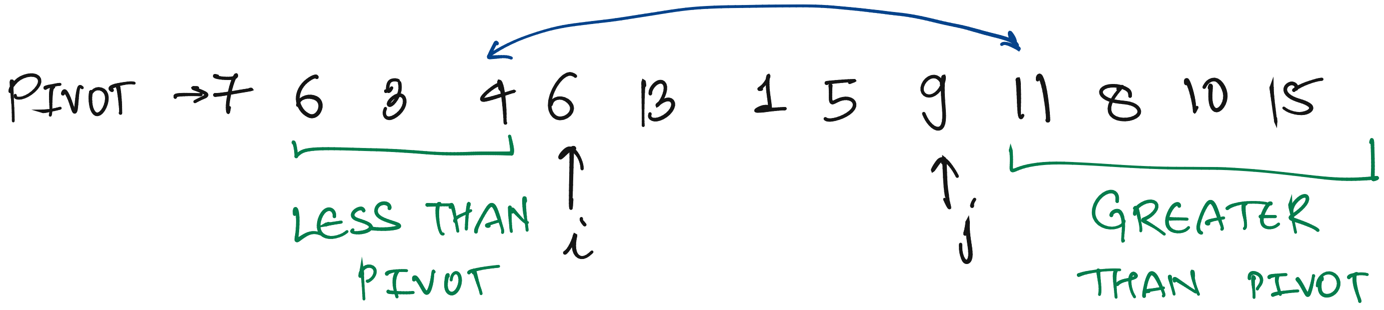
① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT

FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \ldots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j+1 \ldots n]$ ARE GREATER THAN PIVOT.



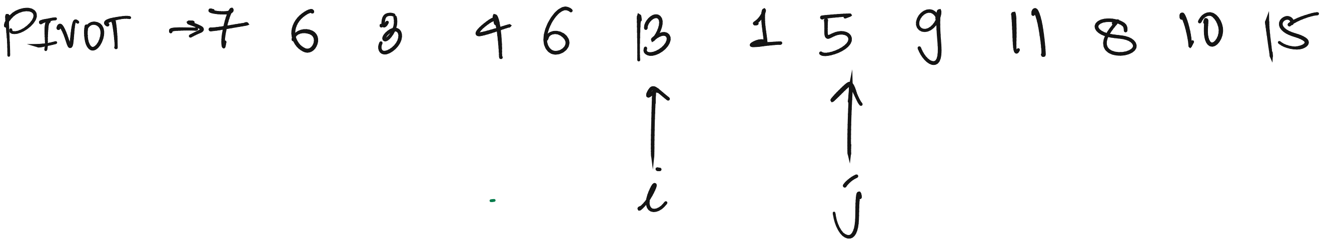
① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT

FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \ldots i\text{-}1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j{+}1 \ldots n]$ ARE GREATER THAN PIVOT.



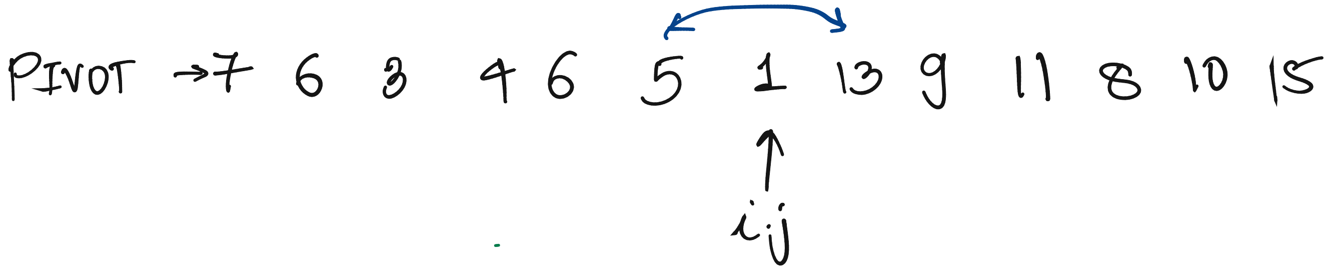
① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT

FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \ldots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j+1 \ldots n]$ ARE GREATER THAN PIVOT.



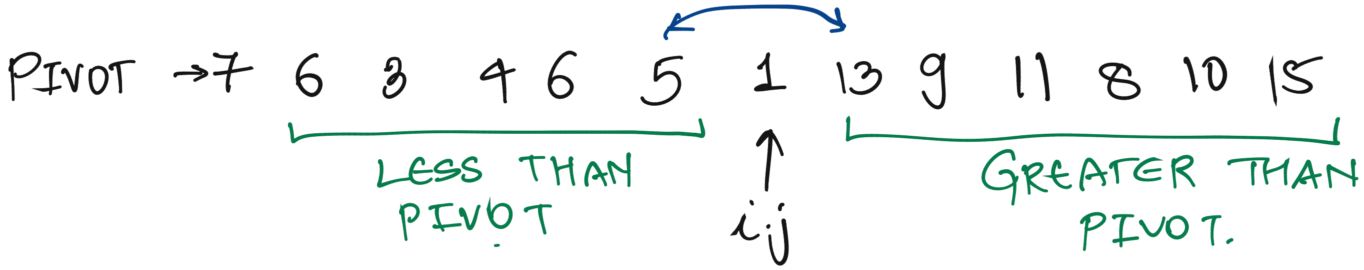
① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT

FEW OBSERVATIONS :

1) NUMBERS IN A[2... i-1] ARE LESS THAN PIVOT

2) NUMBERS IN A[j+1... n] ARE GREATER THAN PIVOT.



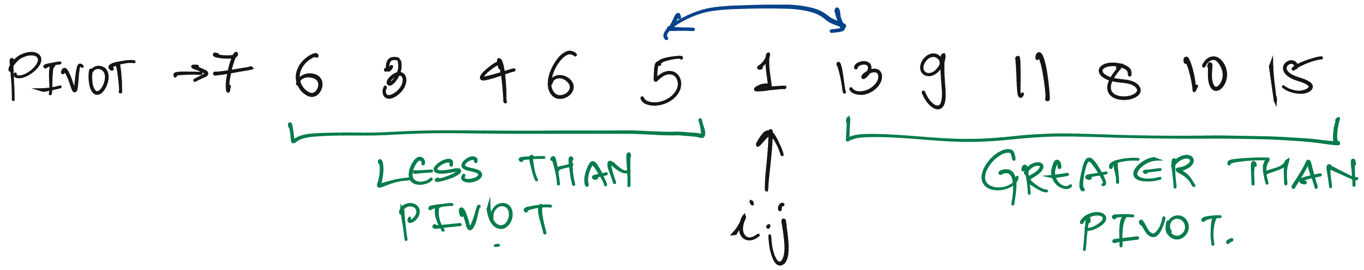
① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT

FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \ldots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j+1 \ldots n]$ ARE GREATER THAN PIVOT.



1. INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT TILL YOU MEET j
2. DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT TILL YOU MEET i

FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \ldots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j+1 \ldots n]$ ARE GREATER THAN PIVOT.

PIVOT → 7 6 3 4 6 5 1 13 9 11 8 10 15

↑

i,j

① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT TILL YOU MEET j

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT TILL YOU MEET i

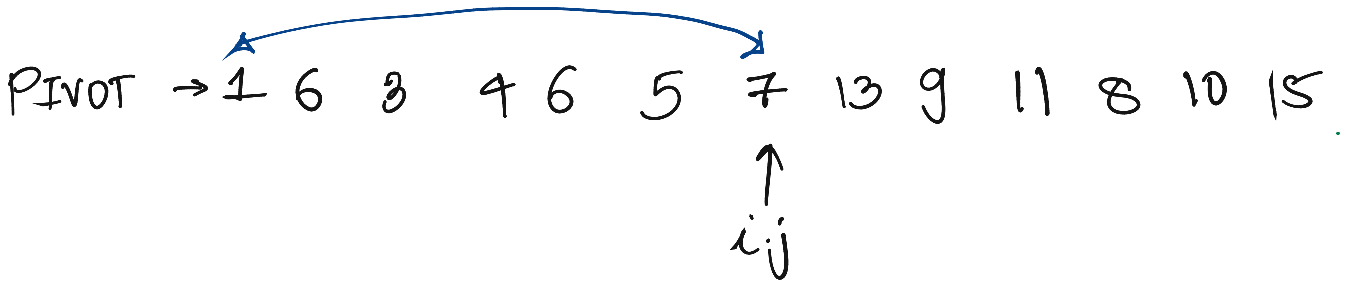
FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \ldots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j+1 \ldots n]$ ARE GREATER THAN PIVOT.

LAST STEP (WHEN $i=j$)

```
IF A[i] < PIVOT
    SWAP ( A[i] , A[1] )
```



① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT TILL YOU MEET j

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT TILL YOU MEET i

FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \ldots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j+1 \ldots n]$ ARE GREATER THAN PIVOT.

LAST STEP (WHEN $i=j$)

```
IF A[i] < PIVOT
    SWAP ( A[i] , A[1] )
```

PIVOT → 7 6 3 4 6 5 20 13 9 11 8 10 15

↑

i,j

① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT TILL YOU MEET j

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT TILL YOU MEET i

FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \dots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j+1 \dots n]$ ARE GREATER THAN PIVOT.

LAST STEP (WHEN $i=j$)

IF $A[i] <$ PIVOT

SWAP ($A[i]$, $A[1]$)

PIVOT → 7 6 3 4 6 5 20 13 9 11 8 10 15

↑

i,j

① INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT TILL YOU MEET j

② DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT TILL YOU MEET i

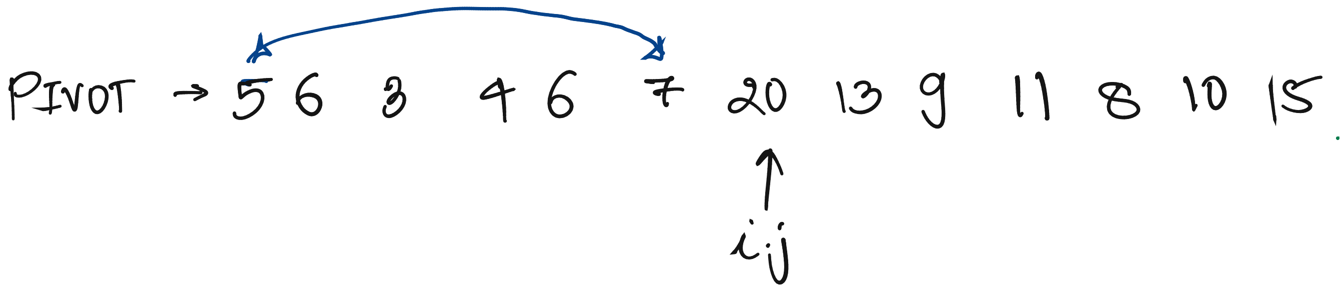
FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \ldots i-1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j+1 \ldots n]$ ARE GREATER THAN PIVOT.

LAST STEP (WHEN $i = j$)

```
IF  A[i] < PIVOT
        SWAP ( A[i] , A[1] )
ELSE
        SWAP ( A[i-1], A[1] )
```



(1) INCREMENT i TILL YOU FIND A NUMBER GREATER THAN PIVOT TILL YOU MEET j

(2) DECREMENT j TILL YOU FIND A NUMBER LESS THAN PIVOT TILL YOU MEET i

FEW OBSERVATIONS :

1) NUMBERS IN $A[2 \dots i\text{-}1]$ ARE LESS THAN PIVOT

2) NUMBERS IN $A[j{+}1 \dots n]$ ARE GREATER THAN PIVOT.

LAST STEP (WHEN $i = j$)

```
IF A[i] < PIVOT
        SWAP ( A[i] , A[1] )
ELSE
        SWAP ( A[i-1], A[1] )
```

```
PIVOT ← A[1];
i ← 2 ;   j ← n
WHILE (TRUE)
{   WHILE (A[i] < PIVOT AND i < j)
            i ← i+1;
    WHILE ( A[j] > PIVOT AND j > i)
            j ← j-1;
    IF  i = j
          BREAK;
    ELSE
    {       SWAP (A[i], A[j])
            i ← i+1;
            j ← j+1;
    }
}
IF ( A[i] < PIVOT)
      SWAP ( A[i], A[1])
ELSE
      SWAP (A[i-1], A[1])
```

```
PIVOT ← A[1];
i ← 2;   j ← n
WHILE (TRUE)
{   WHILE (A[i] < PIVOT AND i < j)
            i ← i+1;
    WHILE (A[j] > PIVOT AND j > i)
            j ← j-1;
    IF   i = j
          BREAK;
    ELSE
    {       SWAP (A[i], A[j])
            i ← i+1;
            j ← j+1;
    }
}
IF ( A[i] < PIVOT)
      SWAP (A[i], A[1])
ELSE
      SWAP (A[i-1], A[1])
```

RUNNING TIME = $O(n)$ BUT ONLY ONE PASS OVER THE ARRAY.

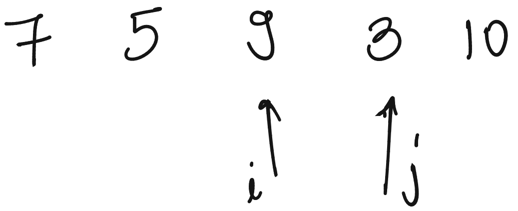
```
PIVOT ← A[1] ;
 i ← 2 ;   j ← n
 WHILE (TRUE)
 {   WHILE (A[i] < PIVOT  AND  i < j)
             i ← i+1 ;
     WHILE ( A[j] > PIVOT  AND  j > i)
             j ← j-1 ;
     IF    i = j
           BREAK ;
     ELSE
     {       SWAP (A[i] , A[j])
              i ← i+1 ;
              j ← j+1;
     }
 }
 IF ( A[i] < PIVOT)
       SWAP ( A[i], A[1])
 ELSE
       SWAP (A [i-1], A[1])
```

7 5 9 3 10

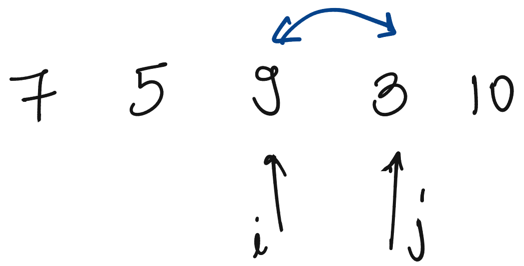
```
PIVOT ← A[1] ;
 i ← 2 ;   j ← n
 WHILE (TRUE)
 {   WHILE (A[i] < PIVOT  AND  i < j)
             i ← i+1 ;
     WHILE (A[j] > PIVOT  AND  j > i)
             j ← j-1 ;
     IF   i = j
           BREAK ;
     ELSE
     {       SWAP (A[i], A[j])
             i ← i+1 ;
             j ← j+1;
     }
 }
 IF ( A[i] < PIVOT)
       SWAP ( A[i], A[1])
 ELSE
       SWAP (A[i-1], A[1])
```

```
  7   5   9   3   10
      ↑           ↑
      i           j
```

```
PIVOT ← A[1];
 i ← 2;   j ← n
 WHILE (TRUE)
 {    WHILE (A[i] < PIVOT  AND  i < j)
            i ← i+1;
      WHILE (A[j] > PIVOT  AND  j > i)
            j ← j-1;
      IF   i = j
           BREAK;
      ELSE
      {       SWAP (A[i], A[j])
              i ← i+1;
              j ← j+1;
      }
 }
 IF ( A[i] < PIVOT)
      SWAP (A[i], A[1])
 ELSE
      SWAP (A[i-1], A[1])
```



```
PIVOT ← A[1];
 i ← 2 ;   j ← n
 WHILE (TRUE)
 {    WHILE (A[i] < PIVOT  AND  i < j)
             i ← i+1 ;
      WHILE ( A[j] > PIVOT  AND  j > i)
             j ← j-1 ;
      IF     i = j
            BREAK ;
      ELSE
      {       SWAP (A[i] , A[j])
               i ← i+1 ;
               j ← j+1;
      }
 }
 IF ( A[i] < PIVOT)
        SWAP ( A[i] , A[1])
 ELSE
        SWAP (A [i-1], A[1])
```



```
PIVOT ← A[1] ;
 i ← 2 ;   j ← n
 WHILE (TRUE)
 {    WHILE (A[i] < PIVOT  AND  i < j)
            i ← i+1 ;
      WHILE ( A[j] > PIVOT  AND  j > i)
            j ← j-1 ;
      IF   i = j
            BREAK ;
      ELSE
      {      SWAP (A[i] , A[j])
              i ← i+1 ;
              j ← j+1;
      }
 }
 IF ( A[i] < PIVOT)
       SWAP ( A[i] , A[1])
 ELSE
       SWAP ( A [i-1] , A[1])
```

```
7   5   3   9   10
        ↑j  ↑i
```

```
PIVOT ← A[1];
 i ← 2 ;   j ← n
 WHILE (TRUE)
 {   WHILE (A[i] < PIVOT  AND  i < j)
          i ← i+1 ;
     WHILE ( A[j] > PIVOT  AND  j > i)
          j ← j-1 ;
     IF   i ≤ j
          BREAK ;
     ELSE
     {       SWAP (A[i] , A[j])
              i ← i+1 ;
              j ← j+1;
     }
 }
 IF ( A[i] < PIVOT)
      SWAP ( A[i] , A[1])
 ELSE
      SWAP (A [i-1] , A[1])
```

```
7   5   3   9   10
        ↑j  ↑i
```

PROBLEM : ASSUME THAT AN ARRAY OF n NUMBERS CONTAINS ONLY K DISTINCT INTEGERS. SORT THE ARRAY USING ONLY $O(1)$ EXTRA MEMORY IN $O(nk)$ TIME.

2 2 5 5 1 1 2 5 2 1

PROBLEM: ASSUME THAT AN ARRAY OF n NUMBERS CONTAINS ONLY K DISTINCT INTEGERS. SORT THE ARRAY USING ONLY $O(1)$ EXTRA MEMORY IN $O(nk)$ TIME.

2 2 5 5 $\underline{1}$ 1 2 5 2 1

FIND THE MINIMUM NUMBER IN $O(n)$ TIME

PROBLEM : ASSUME THAT AN ARRAY OF n NUMBERS CONTAINS ONLY K DISTINCT INTEGERS. SORT THE ARRAY USING ONLY $O(1)$ EXTRA MEMORY IN $O(nk)$ TIME.

2 2 5 5 $\underline{1}$ 1 2 5 2 1

FIND THE MINIMUM NUMBER IN $O(n)$ TIME

1 2 5 5 2 1 2 5 2 1

PROBLEM: ASSUME THAT AN ARRAY OF n NUMBERS CONTAINS ONLY K DISTINCT INTEGERS. SORT THE ARRAY USING ONLY $O(1)$ EXTRA MEMORY IN $O(nk)$ TIME.

2 2 5 5 $\underline{1}$ 1 2 5 2 1

FIND THE MINIMUM NUMBER IN $O(n)$ TIME

PIVOT → 1 2 5 5 2 1 2 5 2 1

USE THE PARTITION PROCEDURE OF QUICKSORT

PROBLEM: ASSUME THAT AN ARRAY OF n NUMBERS CONTAINS ONLY K DISTINCT INTEGERS. SORT THE ARRAY USING ONLY $O(1)$ EXTRA MEMORY IN $O(nk)$ TIME.

2 2 5 5 1 1 2 5 2 1

FIND THE MINIMUM NUMBER IN $O(n)$ TIME

PIVOT →1 2 5 5 2 1 2 5 2 1

USE THE PARTITION PROCEDURE OF QUICKSORT

1 1 . 1 2 5 5 2 2 5 2

PROBLEM : ASSUME THAT AN ARRAY OF n NUMBERS CONTAINS ONLY K DISTINCT INTEGERS. SORT THE ARRAY USING ONLY $O(1)$ EXTRA MEMORY IN $O(nk)$ TIME.

$2 \quad 2 \quad 5 \quad 5 \quad \underline{1} \quad 1 \quad 2 \quad 5 \quad 2 \quad 1$

FIND THE MINIMUM NUMBER IN $O(n)$ TIME

PIVOT
$\rightarrow 1 \quad 2 \quad 5 \quad 5 \quad 2 \quad 1 \quad 2 \quad 5 \quad 2 \quad 1$

USE THE PARTITION PROCEDURE OF QUICKSORT

$\underbrace{1 \quad 1 . 1}_{\text{SORTED}} \quad 2 \quad 5 \quad 5 \quad 2 \quad 2 \quad 5 \quad 2$

PROBLEM: ASSUME THAT AN ARRAY OF n NUMBERS CONTAINS ONLY K DISTINCT INTEGERS. SORT THE ARRAY USING ONLY $O(1)$ EXTRA MEMORY IN $O(nk)$ TIME.

$$2 \quad 2 \quad 5 \quad 5 \quad \underline{1} \quad 1 \quad 2 \quad 5 \quad 2 \quad 1$$

FIND THE MINIMUM NUMBER IN $O(n)$ TIME

PIVOT $\rightarrow$ $1 \quad 2 \quad 5 \quad 5 \quad 2 \quad 1 \quad 2 \quad 5 \quad 2 \quad 1$

$$\left.\begin{array}{l}\text{USE THE PARTITION PROCEDURE OF}\\ \text{QUICKSORT}\end{array}\right\} O(n)$$

$$\underbrace{1 \quad 1 \, . \, 1}_{\text{SORTED}} \quad 2 \quad 5 \quad 5 \quad 2 \quad 2 \quad 5 \quad 2$$

| ← REPEAT HERE → |

RUNNING TIME = $O(nk)$

PROBLEM : ASSUME THAT AN ARRAY OF n NUMBERS CONTAINS ONLY K DISTINCT INTEGERS. SORT THE ARRAY USING ONLY $O(1)$ EXTRA MEMORY IN $O(nk)$ TIME.

2 2 5 5 $\underline{1}$ 1 2 5 2 1

FIND THE MINIMUM NUMBER IN $O(n)$ TIME

PIVOT → 1 2 5 5 2 1 2 5 2 1

USE THE PARTITION PROCEDURE OF QUICKSORT

$\underbrace{1 \quad 1 . 1}_{\text{SORTED}}$ 2 5 5 2 2 5 2

|← REPEAT HERE →|