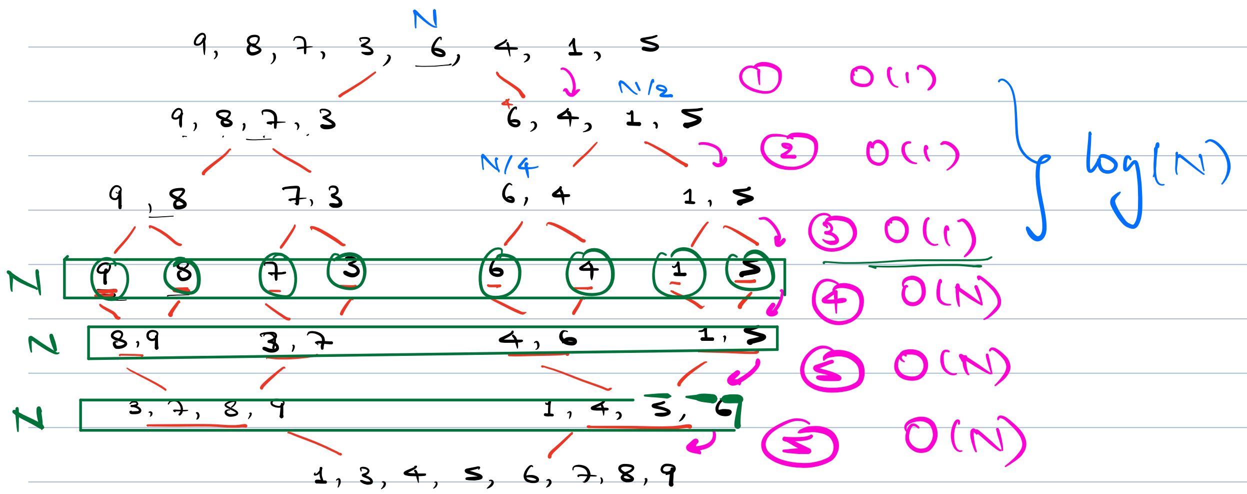


$$T.C. = O(1 \times \log(N) + N\log(N))$$

$$= O(N\log N)$$

Recursive relation

$$T(N) = 2 \times T(N/2) + N$$

Refer to intermediate

Q Given 2 arrays $A[N]$ & $B[M]$

Count the no. of pairs i, j s.t.

$A[i] > B[j]$.

Eg $A: \begin{array}{ccc} 0 & 1 & 2 \\ 7, & 3, & 5 \end{array}$

✓ B : $\begin{array}{ccc} 2, & 0, & 6 \\ 0 & 1 & 2 \end{array}$

(7, 2) (7, 0) (7, 6)
(3, 2) (3, 0) (5, 2) ⇒ $\boxed{7}$
(5, 0)

$$\begin{array}{lcccc} & 0 & 1 & \overset{\Downarrow}{2} & 3 \\ A: & \{2, & 4, & 4, & 5\} \\ B: & \{3, & 2, & 9\} & \\ & 0 & 1 & 2 & \end{array}$$

$\left.\begin{array}{ll} (\overset{1}{4}, \overset{0}{3}) & (\overset{1}{4}, \overset{1}{2}) \\ (\underset{2}{4}, \underset{0}{3}) & (\underset{2}{4}, \underset{1}{2}) \\ (\underset{3}{5}, \underset{0}{3}) & (\underset{3}{5}, \underset{1}{2}) \end{array}\right\}$ $\underline{\underline{6}}$

Solⁿ ① Brute force

for every element in A, we check every element in B & increase count.

T.C. = O(N×M)

Obser

$$\begin{array}{lccc} & \overset{i}{0} & 1 & \overset{i}{2} \\ A: & \{7, & 3, & 5\} \\ B: & \{2, & 0, & \cancel{6}\} \\ & 0 & 1 & \underset{j}{2} \end{array}$$

7 → 3

Sorting arrays.

$\frac{c}{1}$

$A : \{ 3, \cancel{5}, \cancel{7} \}$

$B : \{ 0, 2, \cancel{6} \}$

i	j	cond.	ans
2	2	A[i] > B[j]	2 - 0 + 1
1	2	A[i] < B[j]	0
1	1	A[i] > B[j]	1 - 0 + 1
0	1	A[i] > B[j]	1 - 0 + 1
			$\boxed{7}$

$$T.C. = O(N\log N + M\log M + \underline{N} + \underline{M})$$

$$= O(N\log N + M\log M)$$

Q Google
Amazon
fb
Netflix
CodeNation

Given an array of size N. find the inversion count of the array

no. of pairs (i, j) s.t. $i < j$

& $A[i] > A[j]$

Ex $A : \{ \overset{0}{10}, \overset{1}{\underline{3}}, \overset{2}{\underline{8}}, \overset{3}{\underline{15}}, \overset{4}{\underline{6}}, \overset{5}{\underline{12}}, \overset{6}{\underline{2}}, \overset{7}{\underline{18}}, \overset{8}{\underline{7}}, \overset{9}{1} \}$

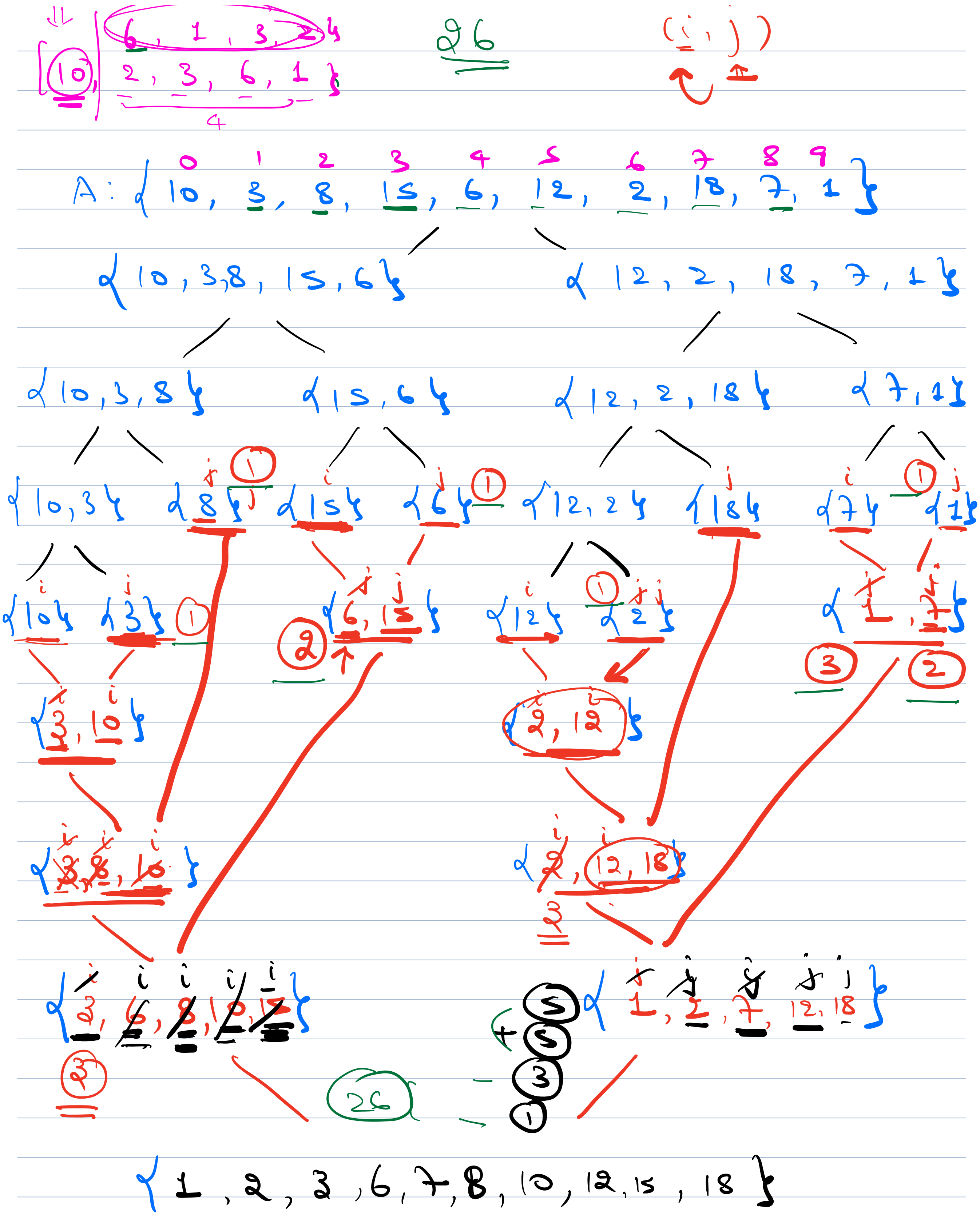
(10, 3)
(10, 8)
(10, 6)
(10, 2)
(10, 7)
(10, 1)

(3, 2)
(3, 1)
(8, 6)
(8, 2)
(8, 7)
(8, 1)

(15, 6)
(15, 12)
(15, 2)
(15, 7)
(15, 1)
(6, 2)

(6, 1)
(12, 2)
(12, 7)
(12, 1)
(2, 1)
(18, 7)

(18, 1)
(7, 1)


10 | 6, 1, 3, 2 }
2, 3, 6, 1 }
4
26
(i , j)
A: { 10, 3, 8, 15, 6, 12, 2, 18, 7, 1 }
0 1 2 3 4 5 6 7 8 9
{ 10, 3, 8, 15, 6 }
{ 12, 2, 18, 7, 1 }
{ 10, 3, 8 }
{ 15, 6 }
{ 12, 2, 18 }
{ 7, 1 }
{ 10, 3 }
{ 8 }
{ 15 }
{ 6 }
{ 12, 2 }
{ 18 }
{ 7 }
{ 1 }
1
1
1
{ 10 }
{ 3 }
1
{ 6, 15 }
2
{ 12 }
{ 2 }
1
{ 1, 7 }
3
2
{ 3, 10 }
{ 2, 12 }
{ 3, 8, 10 }
{ 2, 12, 18 }
3
{ 3, 6, 8, 10, 15 }
23
{ 1, 2, 7, 12, 18 }
5
5
3
1
+
26
{ 1, 2, 3, 6, 7, 8, 10, 12, 15, 18 }

Code

```
count = 0;
void merge (A[], s, mid, e) {

    n1 = mid - s + 1
    n2 = e - (mid+1) + 1 = e - mid

    A1[n1],     A2[n2]

        // fill A1 & A2
    i = 0, j = 0, index = s;
    while ( i < n1  &&  j < n2) {

            if (A2[j] < A1[i]) {
                    i  to  (n1 - i)
                  = (n1 - 1) - i + 1 = (n1 - i)
                  count += (n1 - i);
                  A[index] = A2[j];
                  index++;
                  j++;
            }
            else {
                  Same as prev merge
            }
}
```

T.C. = O(N log N)

Break till 10:40

Q Lock & Key Company (No duplicates)

Google

	0	1	2	3	4	5	6
- L :	5,	3,	1,	2,	7,	6,	11
	0	1	2	3	4	5	6
- K :	1,	6,	5,	7,	11,	3,	2

Return 1 : 1 combination of these locks & keys

Const : You cannot compare a lock with another lock

" " " " key with another key

O/P ⇒ L : { 1, 2, 3, 5, 6, 7, 11 } N

K : { 1, 2, 3, 5, 6, 7, 11 }

Soln ① Brute force

For every lock, find the corresponding key.

No extra space allowed apart from recursion.

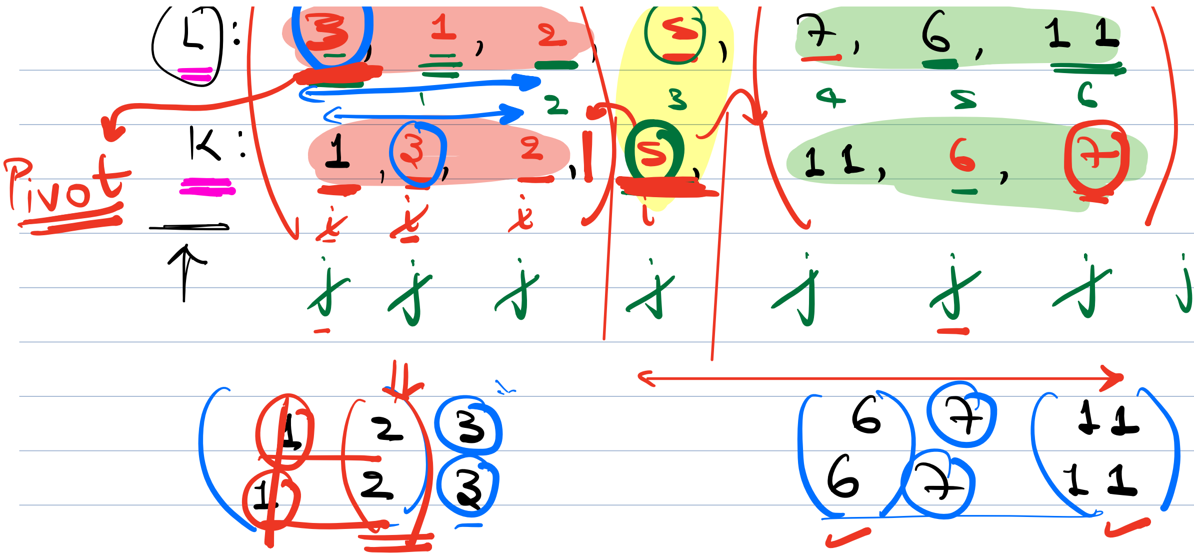
T.C. = $O(N^2)$

lock = 5

j j j j j j j

i i i i

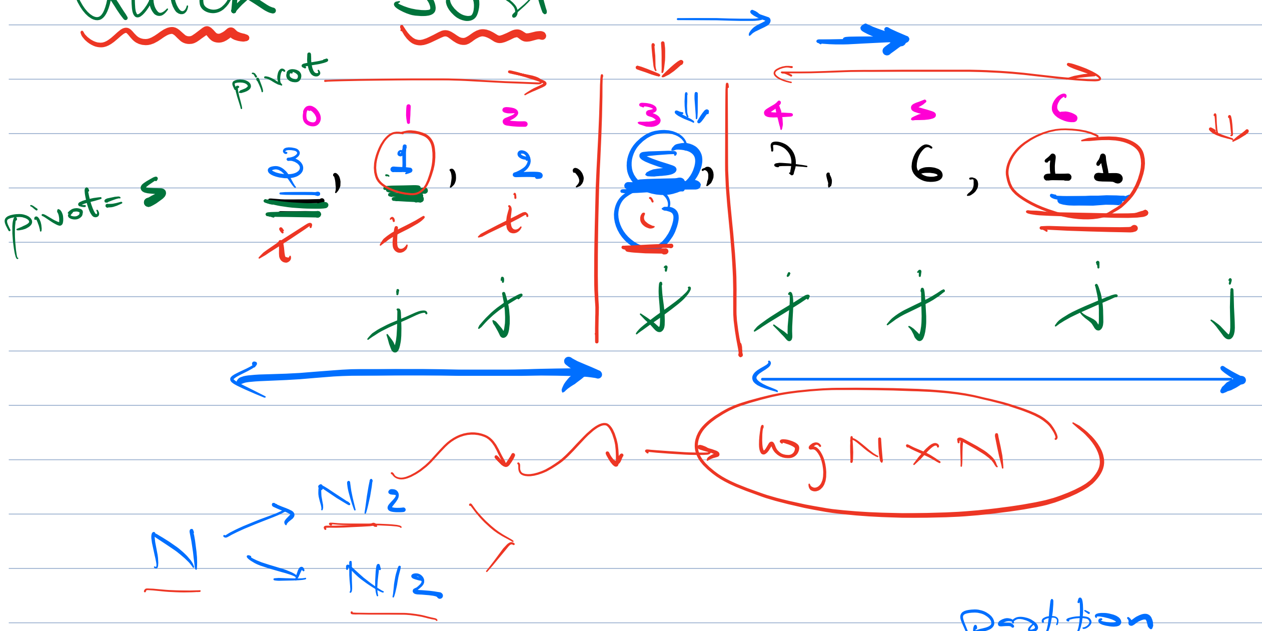
(0 1 2 3 (4 5 6)



1 2 3 5 6 7 11

1 2 3 5 6 7 11

Quick Sort



$$T(N) = 2T(N/2) + \overset{\text{partition}}{N}$$

(5, 3, 1, 2, 7, 6) (11) ×

$$T(N) = T(N-1) + N$$

$$\underbrace{N \to (N-1) \to (N-2) \to (N-3) \dots 1}_{N \times N = O(N^2)}$$

Pivot

$$\underbrace{0\;0\;0\;0\;0\;0\;0\;0}_{N}$$

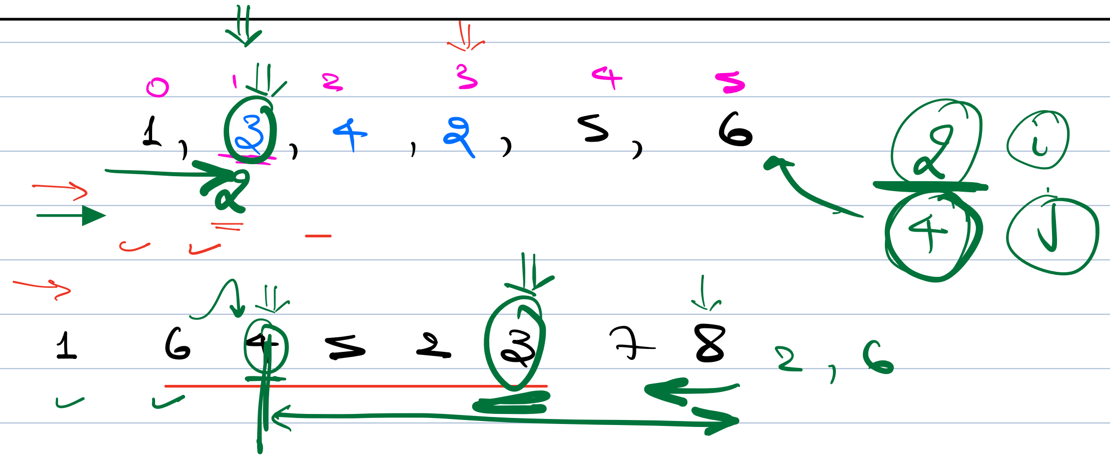
1) first Element
2) Last Element
3) Median
4) Random Element

$$\text{Tim Sort} = \text{Merge} + \text{Insertion}$$

Code

```
void quickSort(A, s, e) {
    // Base Case. H.W.
    pi = partition(arr, s, e); (H.W.)
         1) Selects a pivot (Last)
         2) partition the array
         3) Returns the find index of
            pivot.
    quicksort(A, s, pi-1);
    quicksort(A, pi+1, e);
}
```

Doubts



0 1 2 3 4 5
1, 3, 4, 2, 5, 6
2
2
i
4
j
1 6 4 5 2 3 7 8
2, 6