Merge Sort

→ First we need to know merging.

combining 2 list in 1

e.g

List A	List B
2	5
8	9
15	12
18	17

The list can be of same & different sizes. But, they are sorted.

For merging, we ~~me~~ need another list C

These list can be array / linked list

	A	B	C
i →	2	5 ← j	k
	8	9	
	15	12	
	18	17	

Compare i with j if i is smaller then copy in k

	A	B	C	
	2	5 j	2	
→i	8	9		k
	15	12		
	18	17		

	A	B		C	
	2	5		2	
i→	8	9	←j	5	
	15	12		←	k
	18	17			

Continue until ~~[illegible]~~ all sorted.

in the end.

	A	B	C
			2
	2	5	5
	8	9	8
		12	
	15		9
i→	18	17	12
		←j	15
			17
			←k

if one list is finished, copy remeining items of the list

so C list is with sorted elements

This was merging

Assume

A has m elements
B has n elements
So Totel elements sorted are

$\Theta(m+n)$

Now Algorithm of merge.

```
                              List    size
Algorithm merge ( A , B , m , n )
{
   i=1 , j=1 , k=1            ∵ Assuming index 1 so fer pesudocode.
   while ( i<=m && j<=n)      ↳ Until one list finishes. this will happen
   { if ( A[i] < B[j] )
        c[k] = A[i]  ]
        k++ ; i++    ]→ c[k++] + A[i++]
```

```
else
    c[k++] = B[j++]
}
```

// Once while is done, there might be a possiblity that list have remaing item. We dont know which one so we need to copy all remaining elements (Assume List A have more elements)

```
for( ; i<=m ; i++)
    c[k++] = A[i];
for ( ; j<=n ; j++)
    c[k++] = B[j];
}.
```

⇒ What If we more then 2 list ?

A	B	C	D.
4	3	8	2
6	5	10	4
12	9	16	18

→ After comparison ,

Sorted List.

2
3
4

So this one is 4 way merging
or generalize as
m - way merging

= Mostly 2 way merging.

Now we will sort above one. with 2 way merging.

merge.

A	B	C	D
4	3	8	2
6	5	10	4
12	9	16	18

3	2
4	4
5	8
6	10
9	16
12	18

merge

so take pair & do it again

Can we do it another way ?

merg (A B) then (C) then (D)

or (CD) then B the A

Merge Sort

A

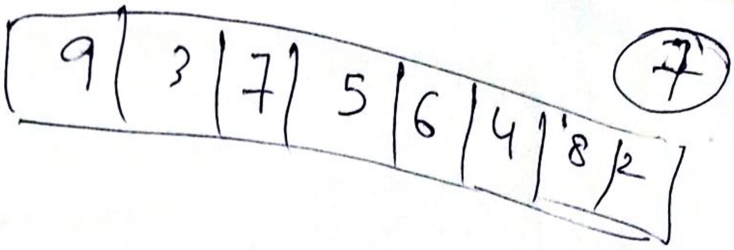
1	2	3	4	5	6	7	8.
9	3	7	5	6	4	8	2

(Done by divide & conquer / recursive

if greter then 1 element, we merge Sort.

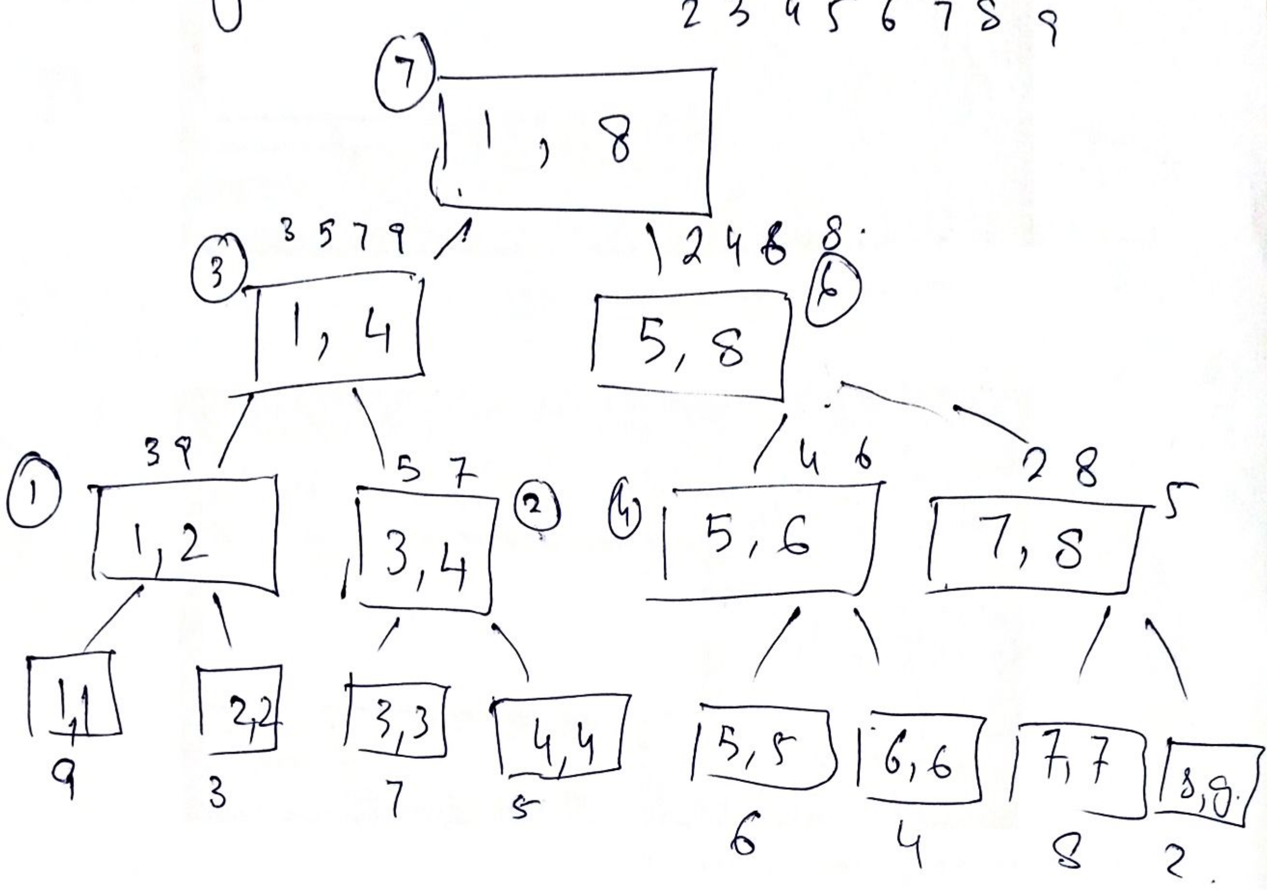
```
                            start    end
                              ↑       ↑
Algorithm Merge Sort (        l,      h      )
{
    if (l < h)
    {
        mid = (l+h)/2
        merge Sort (l, mid)
        merge Sort (mid+1, h);
                   A
        merge ( l, mid, h)
                     B
    }
}
```

Q:- now



low = 1

high = 8.



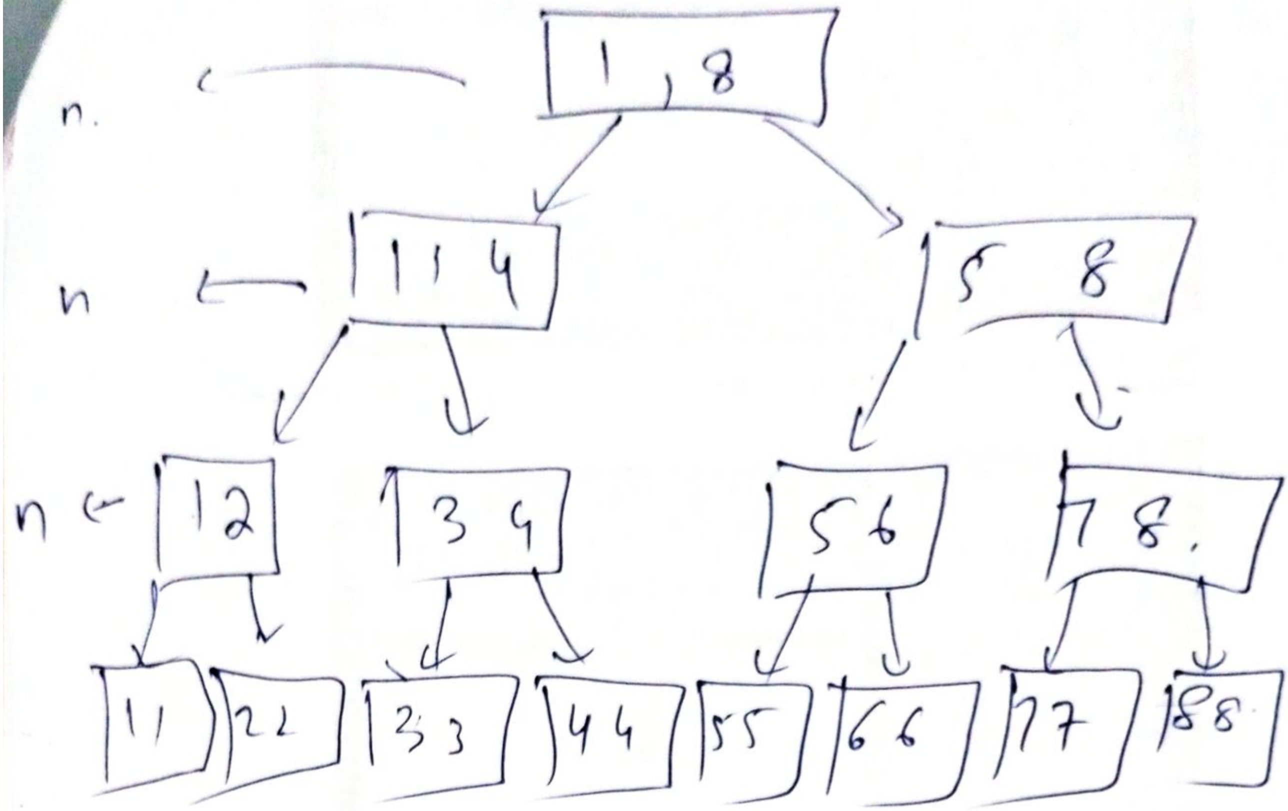
① → First merge

② merge.

... ⑥

So divide & conquer

Now Time Analysis



How many elements merged.

Last layer is only split element

How many levels ~~of~~ for 8 elements

= n so, ~~nlogn~~ (nlogn)

~~Algg~~ Algo Complexity

2

$T(n)$ — Algo merge Sort(l, h)

if (l < h)

mid (l < h)

~~T(n/2)~~ mid = $\frac{(l+h)}{2}$.

1 —

$T(\frac{n}{2})$ / merge Sort (l, mid)

$T(\frac{n}{2})$ " (mid+1, h)

n merge (l mid h)

$$T(n) = \begin{cases} 1 & n = 1 \\ 2\left(\frac{T(n)}{2}\right) + n. & n > 1 \end{cases}$$

master theam. 10

$a = 2$ $\log_b a = \log_2 2 = 1$

$b = 2$

$f(n) = 2$

$k = 1$

$p = 0$

$\log_2 2 = k$

So

$= \Theta(n \log n)$