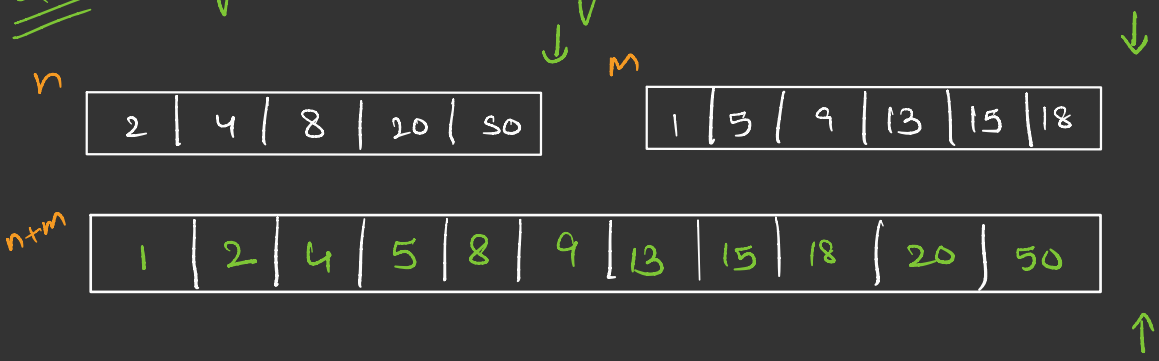
Sorting

Ques Merge 2 sorted arrays



Brute Force

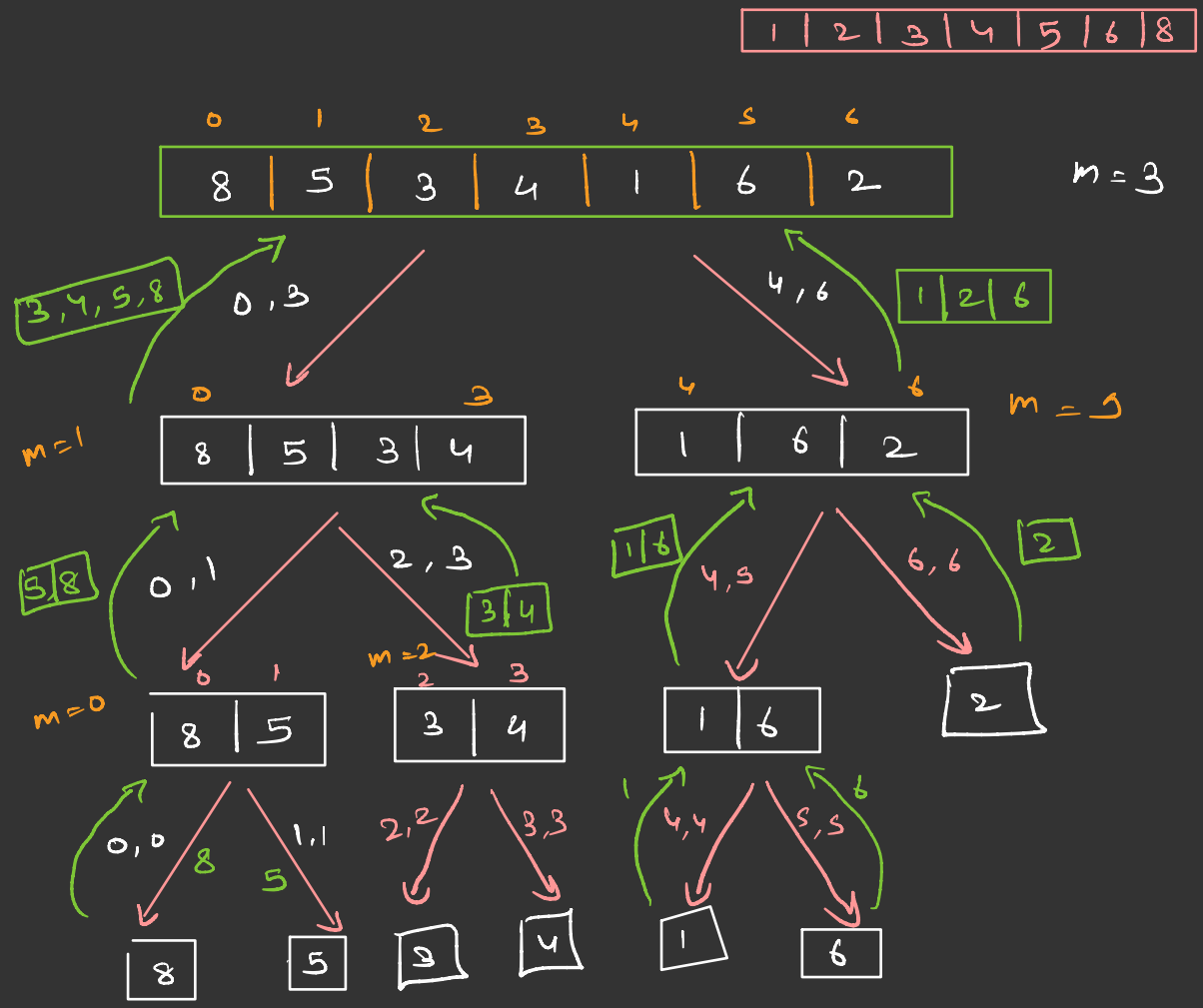
Copy elements from 1st array $O(n)$

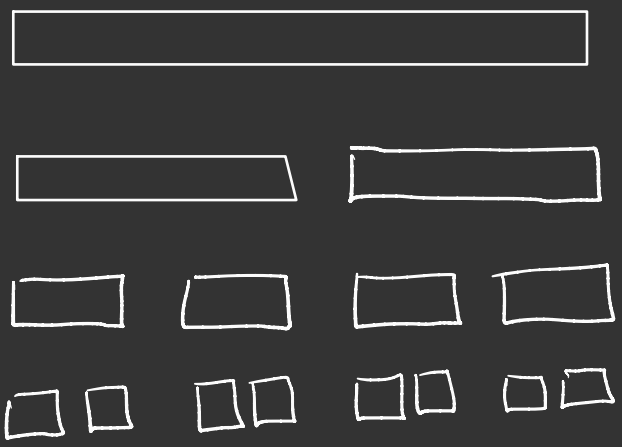
Copy elements from 2nd array $O(m)$

Sort the resultant array $(n+m)\log_2(n+m)$

2 pointer $\Rightarrow$ $O(n+m)$

Merge Sort


1 | 2 | 3 | 4 | 5 | 6 | 8
0 1 2 3 4 5 6
8 | 5 | 3 | 4 | 1 | 6 | 2
m = 3
3, 4, 5, 8
0, 3
4, 6
1 | 2 | 6
m = 1
m = 5
0 3
8 | 5 | 3 | 4
4 6
1 | 6 | 2
5 | 8
0, 1
2, 3
3 | 4
1 | 6
4, 5
6, 6
2
m = 0
m = 2
8 | 5
3 | 4
1 | 6
2
0, 0
1, 1
2, 2
3, 3
4, 4
5, 5
8
5
3
4
1
6



after k th level we got 1 element

$$k = \log_2 n$$

TC $\Rightarrow$ n comparisons on k levels

$\Rightarrow O(n \log_2 n)$

SC $\Rightarrow O(n)$

not inplace sorting

stable sort

Ques Inversion count of the array

8	4	2	1

⇒ 6

(8,4) (4,2)
(8,2) (4,1)
(8,1) (2,1)

arr[i] > arr[j]
i < j

→ how close an array is from being sorted

Sorted → 0

Reverse sorted → max

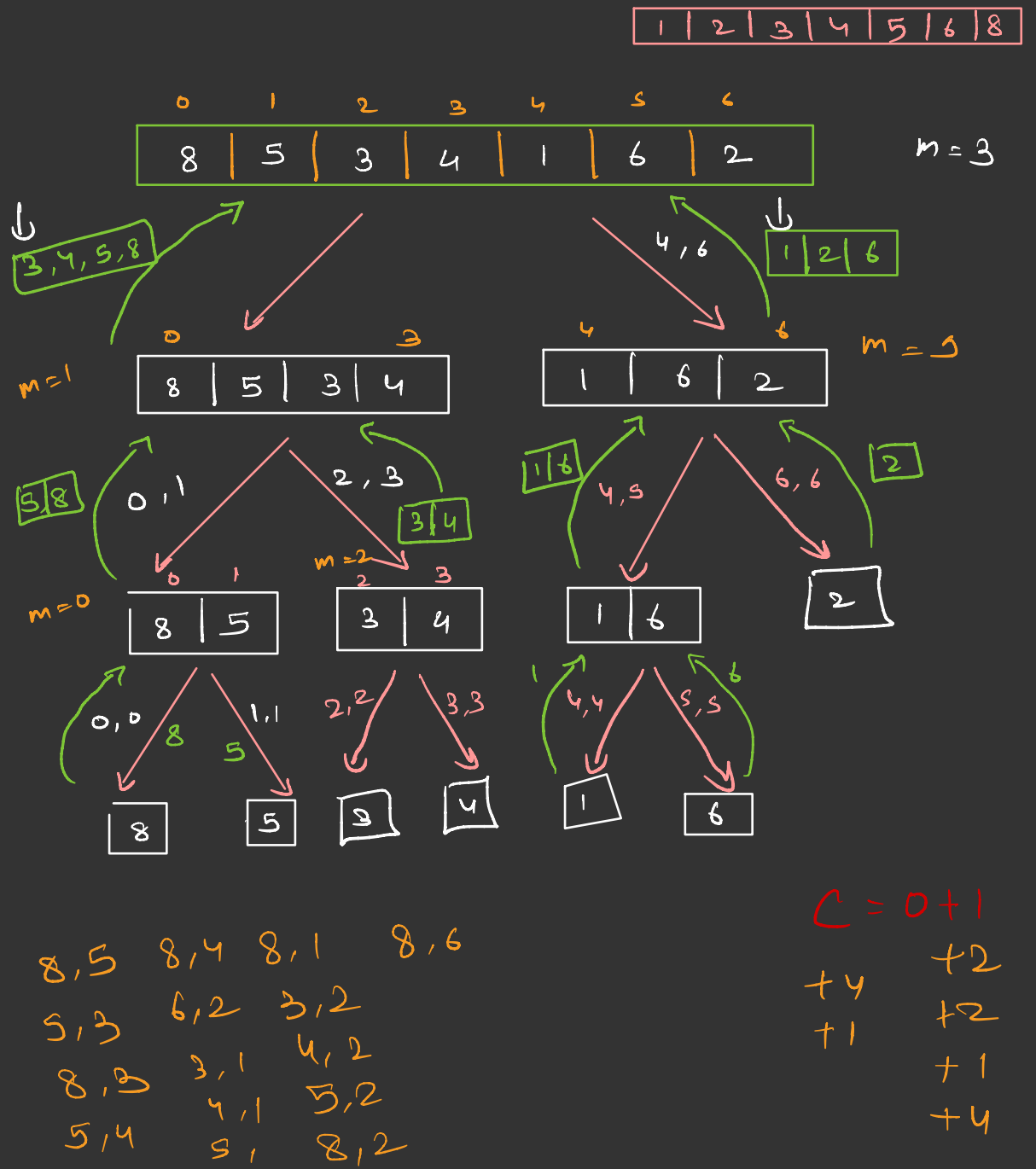
8	5	3	4	1	6	2

Brute Force ⇒ TC ⇒ $O(n^2)$ $c = \phi$

TC ⇒ Merge Sort ⇒ $O(n \log n)$

+4
+2
+1 +2

Merge Sort



8,5 8,4 8,1 8,6
5,3 6,2 3,2
8,3 3,1 4,2
5,4 4,1 5,2
5, 8,2

C = 0 + 1
+4 +2
+1 +2
+1
+4