TCS NQT

Programming Lecture – 3

HashMaps

HashMap

→ Key → Value

'Banana' → 6

Apple → 5

arr = [1, 2, 3, 1, 3, 4, 2]

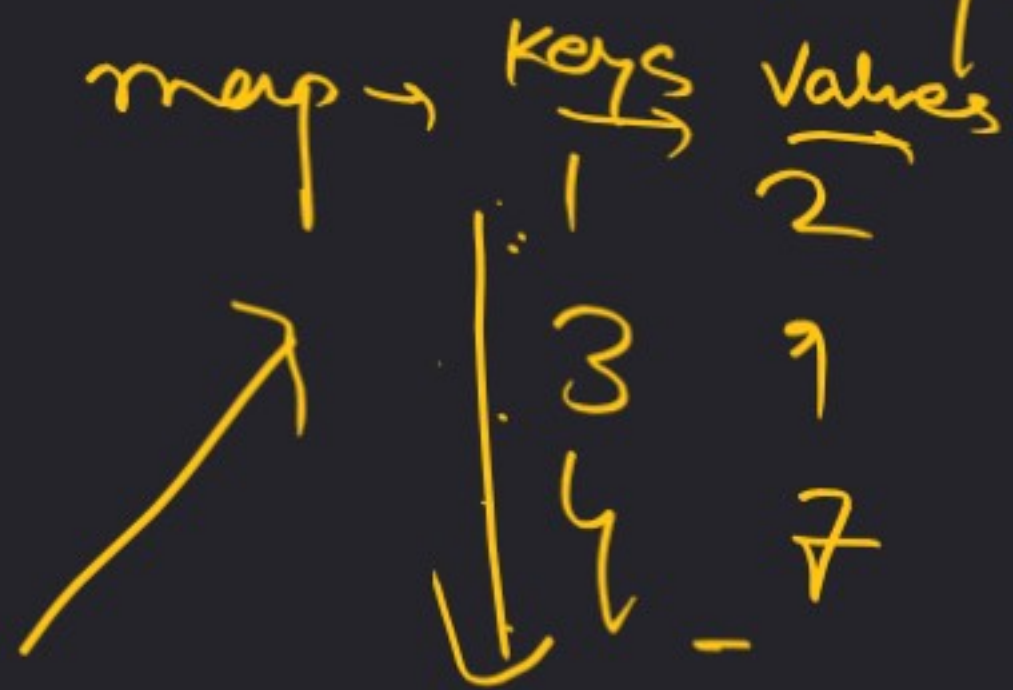
map:

key	value
1	2
2	2
3	2
4	1

map.size() → 4

ordered Map.

↳ keys Sorted.



Ordered Map

map<T, T> mpj

Name of Variable

Type of Key

Type of Value

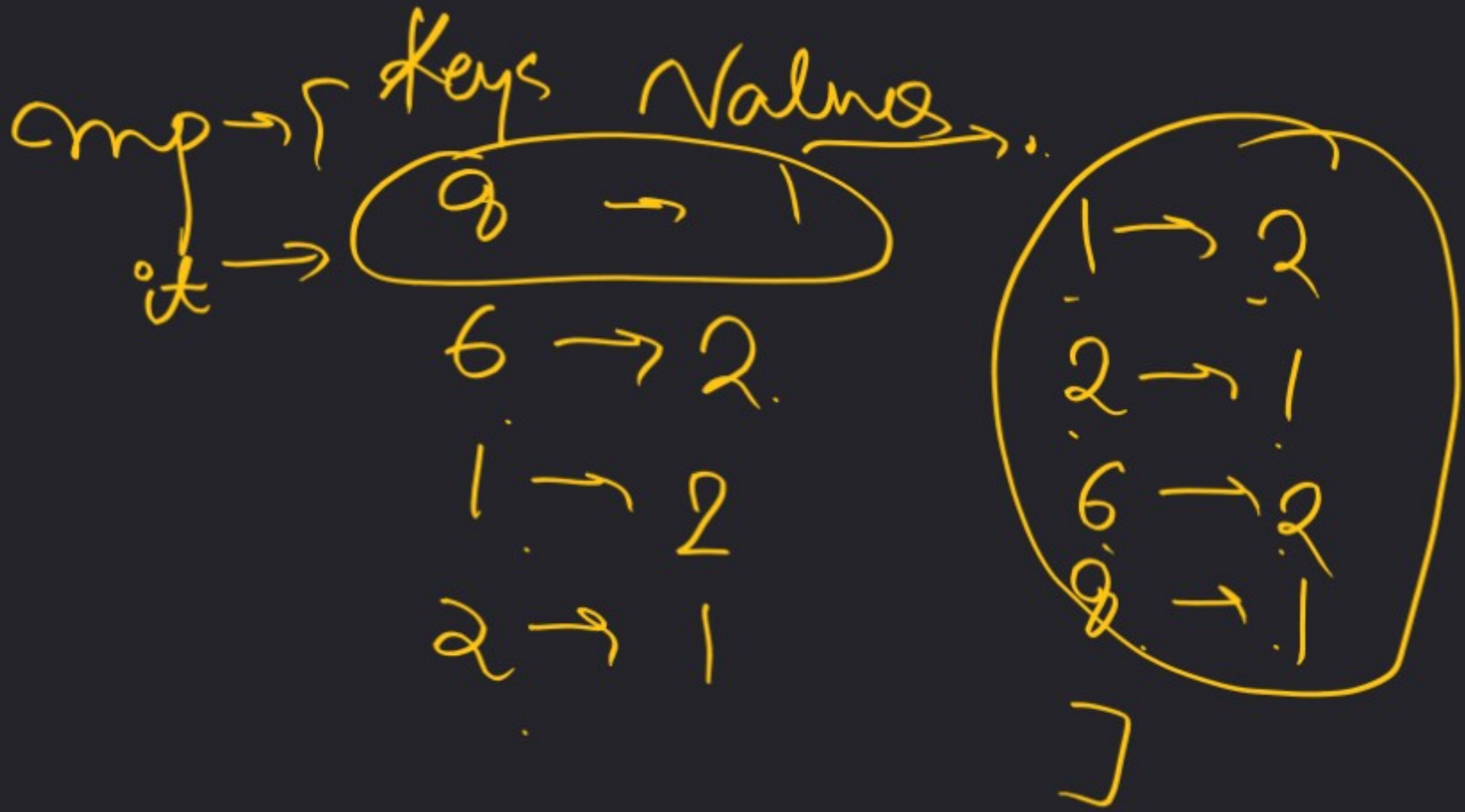
map<int, int> mp;

log(n)

↑ mp.size();

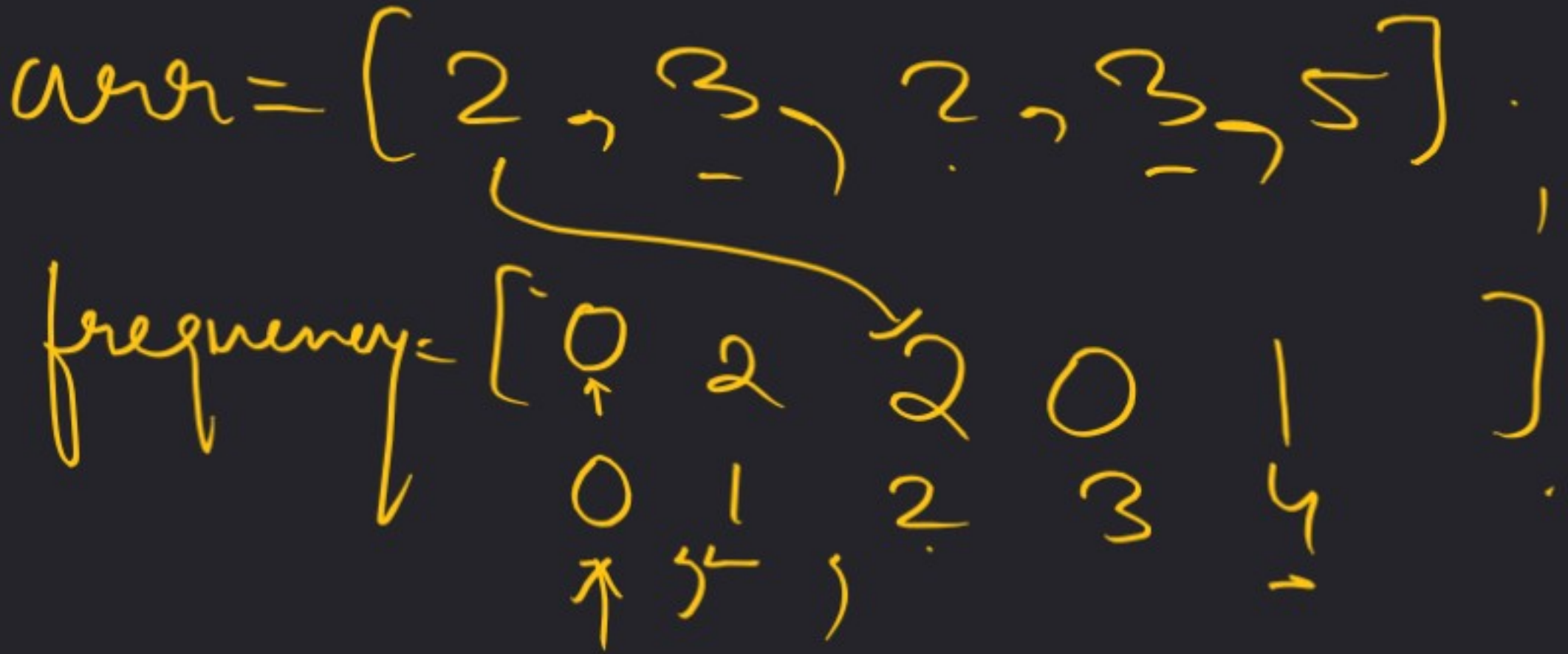
```
arr = [8, 6, 1, 9, 1, 6]

map<int, int> mp;
for(int i=0; i<arr.size(); i++){
    mp[arr[i]]++;
}
```


mp
keys
values
it
9 → 1
6 → 2
1 → 2
2 → 1
1 → 2
2 → 1
6 → 2
9 → 1
]

```
Hashmap<int,int> mp =
        new Hashmap<>();
```

```
for( auto it : mp) {
    int key = it.first;
    int value = it.second;
    cout<<key<<" "<<value<<endl;
}
```


arr = [2, 3, 2, 3, 5]
frequency = [0 2 2 0 1]
0 1 2 3 4

arr = [2 , 1 , 3 , 1 , 2 , 4]

return(3)

arr = [1, 2, 6, 4, 4]

frequency Map →

1 → 1
2 → 1
6 → 1
4 → 2

res → [1, 2, 6, 4]

arr = [3, 3, 1, 2, 2]

ans = [3, 1, 2]

map = { 3 → 2
1 → 1
2 → 2 }

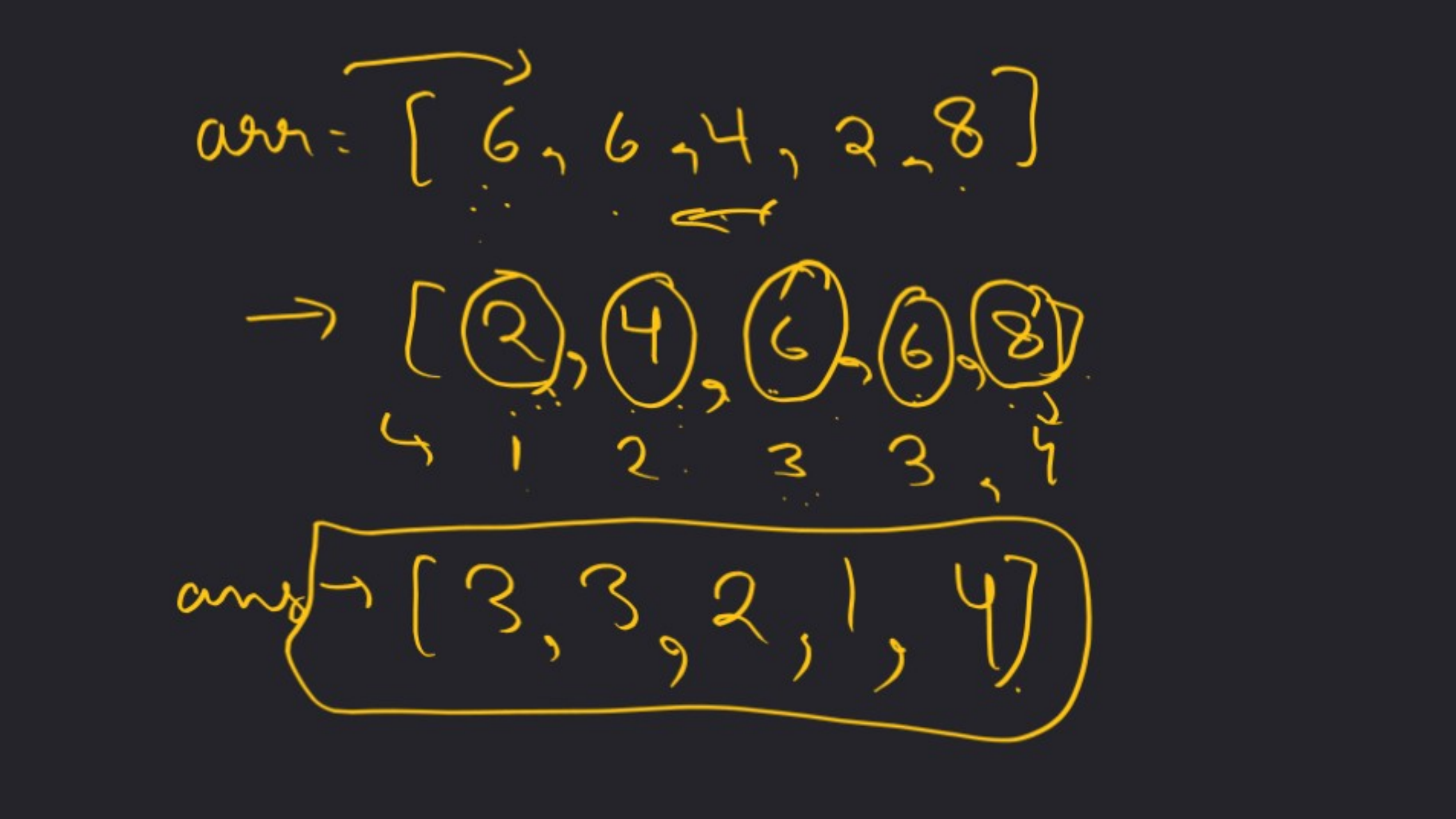
res → [3, 1, 2]

arr = [20, 15, 26, 2, 98, 6]

sort → [2, 6, 15, 20, 26, 98]

0 1 2 3 4 5

res → [4, 3, 5, 1, 6, 2]


arr = [6, 6, 4, 2, 8]
[2, 4, 6, 6, 8]
4 1 2 3 3 4
ans → [3, 3, 2, 1, 4]

rank = 4.

arr = [[[[1], [1]], [[2], [3, 3]]]].

mp[3] → 3.

1 → 1

2 → 2

3 → 3

arr = [2, 3, -2, 6, -10]

SA = [2]

[2, 3]

[2, 3, -2]

[2, 3, -2, 6] = 9.

⇒ [2, 3, 6]

arr :- [-1, -2, -100].

arr = [4 -2 4 -7 , 3 , 4 , -2]

Sum = 7 - 2 = 5

Sum → 5 or -2.

✓

ans = 7

☺

arr = [9, -1, 3, 4]

-1e4 → (10000) ↪ 4

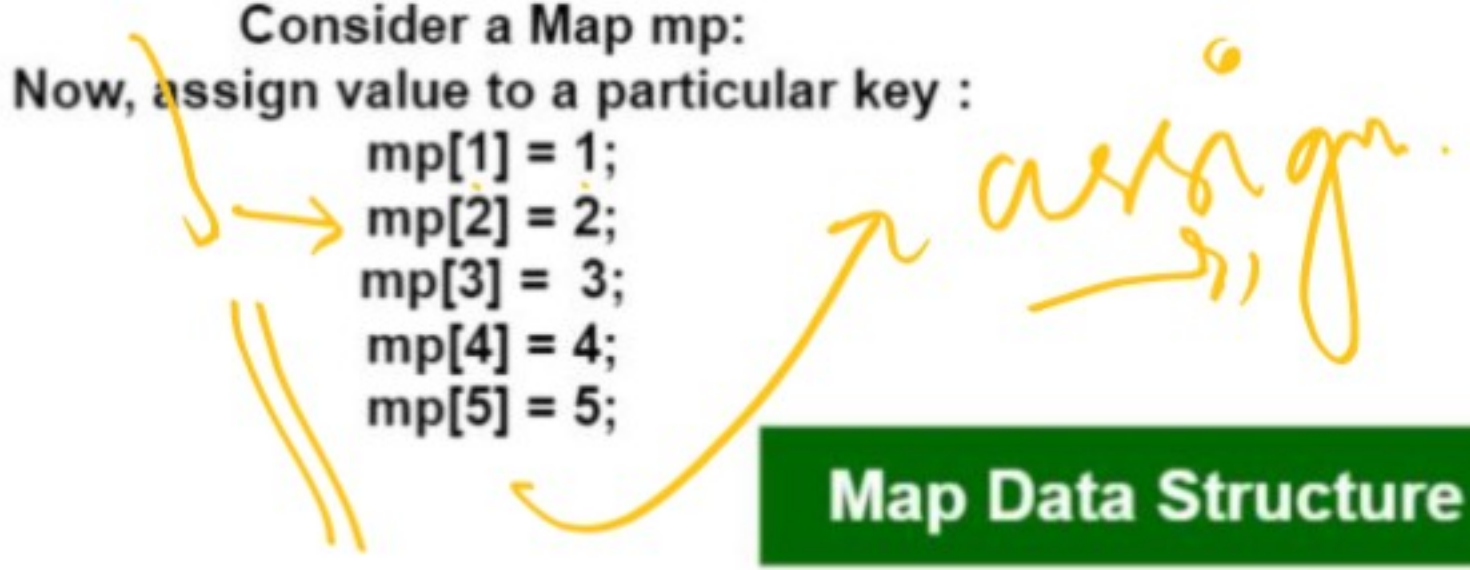
Inserting Elements inside Map Data Structure

Elements:	1	2	3	4	5

Consider a Map mp:

Now, assign value to a particular key :

```
mp[1] = 1;
mp[2] = 2;
mp[3] = 3;
mp[4] = 4;
mp[5] = 5;
```



Map Data Structure

Key Value Pair Table	
Key	value
1	1
2	2
3	3
4	4
5	5

Array = [-2, -3, 4, -1, -2, 1, 5, -3]

Index	Array								Curr_Sum	Max_Sum
i=0	-2	-3	4	-1	-2	1	5	-3	-2	-2
i=1	-2	-3	4	-1	-2	1	5	-3	-3	-2
i=2	-2	-3	4	-1	-2	1	5	-3	4	4
i=3	-2	-3	4	-1	-2	1	5	-3	3	4
i=4	-2	-3	4	-1	-2	1	5	-3	1	4
i=5	-2	-3	4	-1	-2	1	5	-3	2	4
i=6	-2	-3	4	-1	-2	1	5	-3	7	7
i=7	-2	-3	4	-1	-2	1	5	-3	4	7

Maximum Sum Subarray (Kadane's Algorithm)


THANKYOU