Marks of PCM

Tuesday, 9 November 2021 9:57 PM

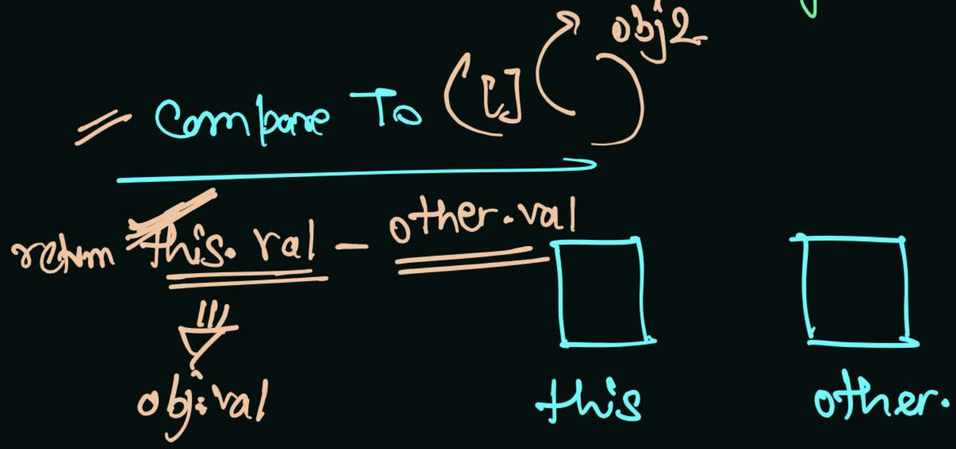
Sort the students.

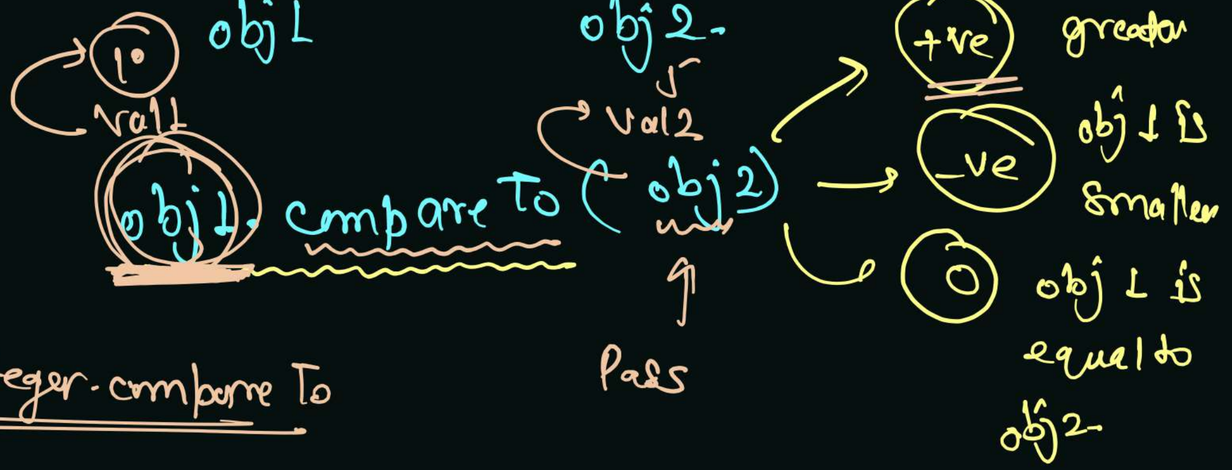
Decision Parameter

(1) Sort on the basis of physics marks. ↑ing order.

(2) If marks in physics of any two students then sort on the basis of chemistry in decreasing order.

(3) If same chem. & phy.. sort on the basis of maths in increasing order.





integer.compareTo

Interface → Comparable

↓

Contract → Method → Declare.

Body - Class in which Comparable interface is implemented.

return this.val - other.val.

```
public int compareTo(ClassName other){
    if(this.phys - other.physics == 0){
        if(this.chemistry - other.chemistry == 0){
            return this.math - other.math;
        } else {
            return other.chimisty - this.chemistry;
        }
    } return this.physic - other.physics;
}
```

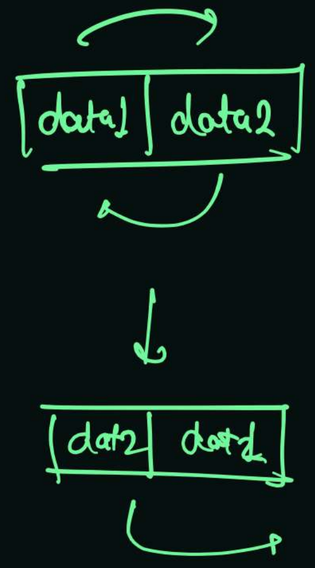
Steps

1. Wrapper class
2. compareTo
3. Code completion

obj1 < obj2

integer/character

```
if( data1 > data2 ) {
    swap(data1, data2);
}
```



integer

```
Car {
    color
    price
    Milage
}
Compareto
```

```
Car [] cars = | car1 | car2. |

if ( car1.price > car2 ) {
    swap(car1, car2);
}
```

val1 — val2

```
Arrays.sort(cars);

[ car1.compareTo( car2) { ]
```

Union of Two Sorted Arrays

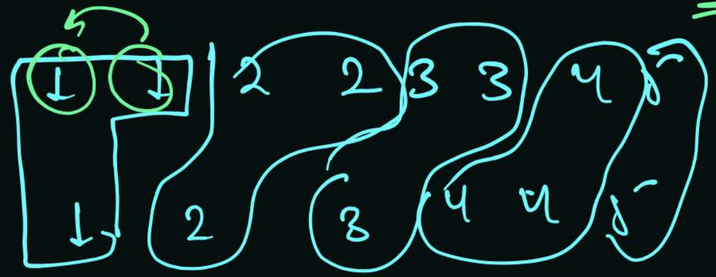
Tuesday, 9 November 2021 9:58 PM

8

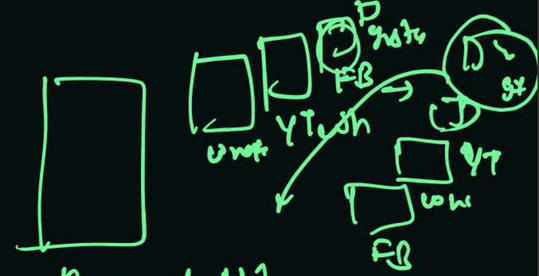
Union of Random array.

arr1 →

arr2 →



HashSet



Recent Apps

D.L
LinkedList + Hashmap

LRU

union →

1 2 3 4 5 6 7

Because given arrays are sorted.

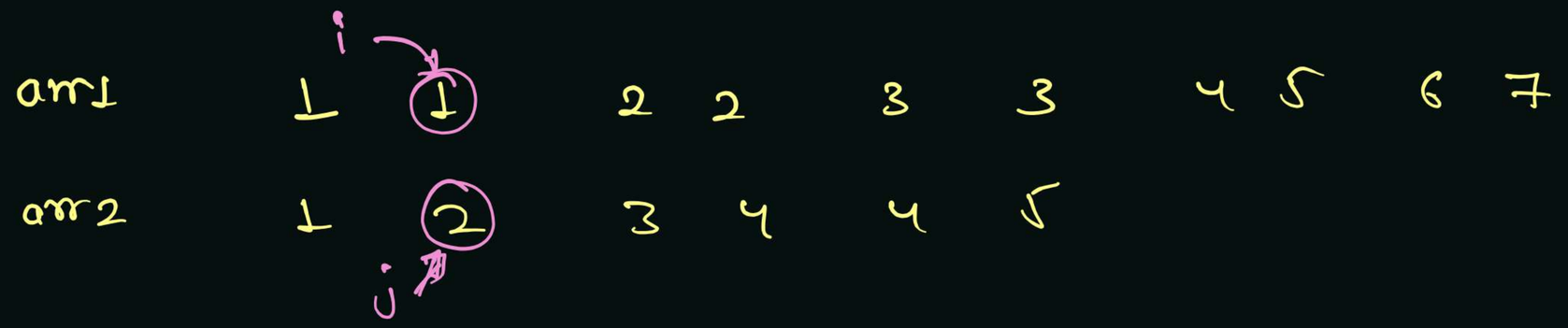
= Brute force ⑧

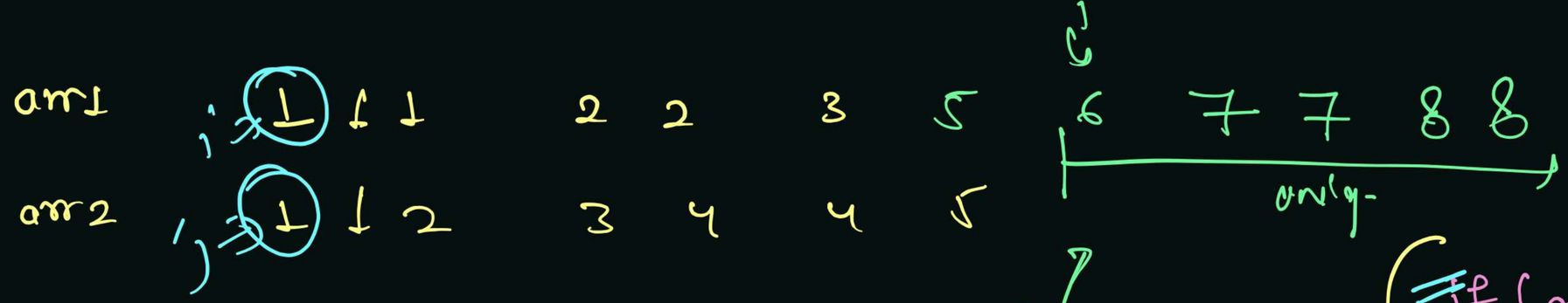
HashSet

= Optimise solution ①

Hint → 2 pointer Approach Similar to merging.

Easy






```
if(list.size() == 0 || list.get(list.size() - 1) != arr[i]
```

list → 1 2 3 4 5

while i & j both are valid;

```
if (arr[i] == arr[j]) {
    if(list.size == 0) {
        list.add(arr[i]);
    } else if(list.get(list.size-1) != arr[i])
        list.add(arr[i]);
    i++; j++;
} else if (arr[i] < arr[j]) {
    if(list.size == 0) {
        list.add(arr[i]);
    } else if (arr[i] != last Element)
        list.add(arr[i]);
    i++;
} else {
    if (list.size == 0)
        list.add(arr[j]);
    else if(arr[j] != last Element)
        list.add(arr[j]);
    j++;
}
```

out of the loop →

add unique Element of i OR j

```
{
=
```

Search A 2D Matrix

Tuesday, 9 November 2021 9:58 PM

Row is sorted -

1	2	3	5	7
10	12	13	15	19
21	23	25	28	29
30	31	35	37	38

First Elemt of current row is greater than last Elemt of previous row.

(1) find row using Binary search

(2) find Elemnt in that row using Binary Search

Total Elemt = m×n

Binary Search = log(m×n)

(12)

Time complexity

normal Approach

O(m+n)

m×n

allowed time complexity

log(m) + log(n)

log(m*n)

find Row index

find

1	2	3	5	7
2	3	8	9	10
7	10	15	16	18
8	12	16	19	20

Row-wise

Column-wise sorted

Box No.

2D

row wise sorted

Column wise sorted.

Data → d

[indx = -1]

lo = 0;

hi = Total Row - 1;

while(lo < hi) {

mid = lo + (hi - lo)/2 ;] mid index

if(arr[mid][0] <= data && arr[mid][total Col - 1] >= data) {

[indx = mid; break;]

} else if(arr[mid][0] > data) {

[hi = mid - 1;]

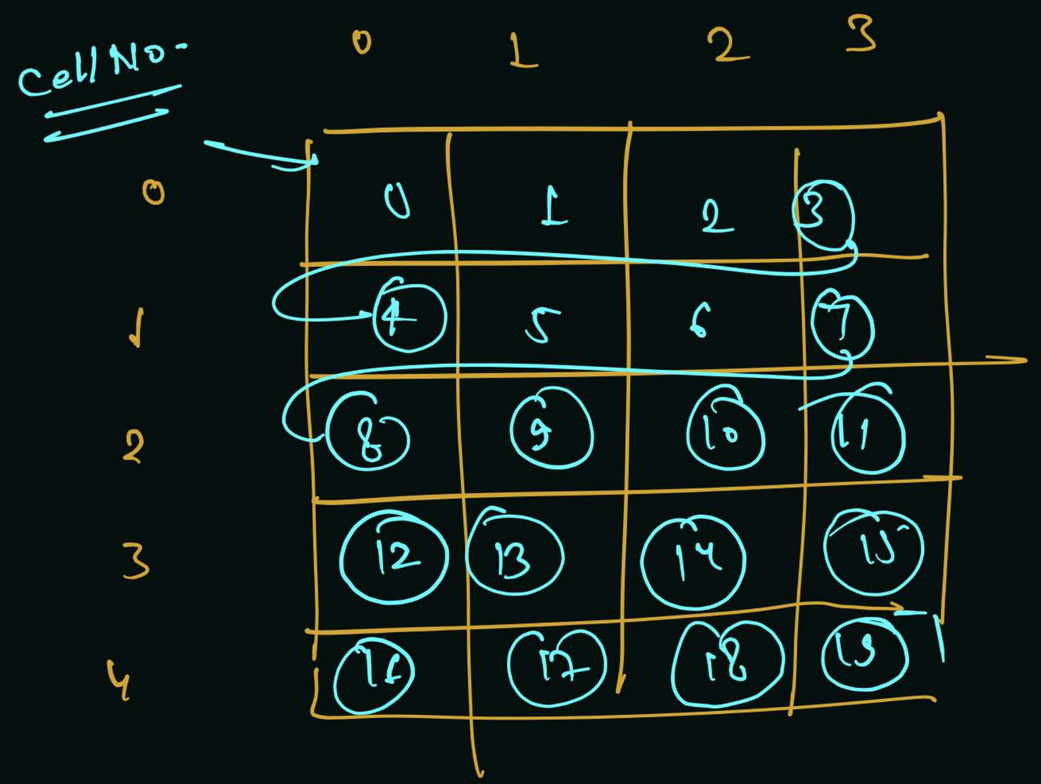
} else {

lo = mid + 1;

}

}

Cell Number. / box Number



Time Complexity

→ log(m * n)

Syllabus

48 topics. → 1 topic → 1 hr.

1 class → 4 hrs. → 4 topics.

→ 1 d + h ld

$\frac{48}{4}$ = (12) class

1 week → 2 class

6 weeks → 1.5 months

Start from -

week - 21st Nov - upto End

Sumeet sir

4 DSA

2 S.D.

Search A 2D Matrix - 2

Tuesday, 9 November 2021 9:59 PM

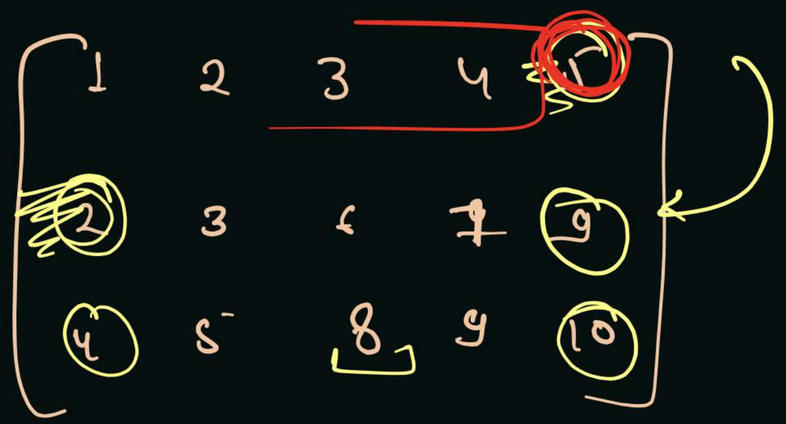
Given → matrix is Row-wise column wise Sorted.

Find an Element

allowed time complexity → $O(m+n)$

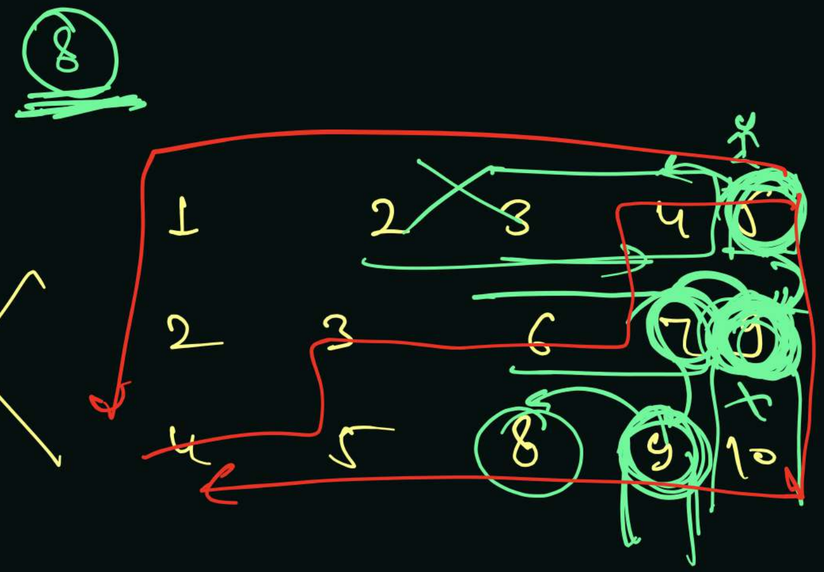
Brute force- $O(m*n)$

why not Binary Search??



find - 8

Using Binary Search ✗

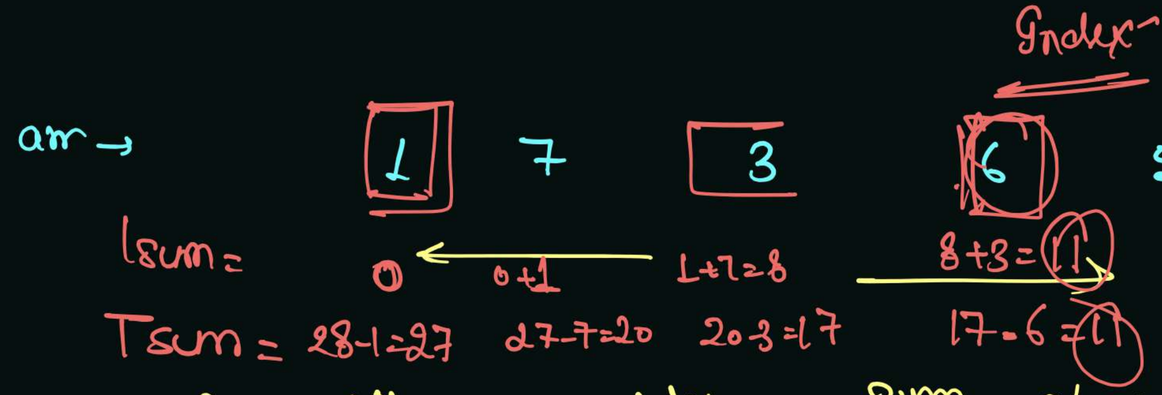


Total travel in x → Total travel in y

$= O(m+n)$

Find Pivot Index

Tuesday, 9 November 2021 9:59 PM



pivot index → index 'i' from which sum of value from 0 to i-1 is equal to sum of value from i+1 to n-1, the 'i' is pivot.

① Brute force

→ T = $O(n^2)$. Space - $O(1)$

② Optimised 1

→ Time - $O(n)$
Space - $O(n)$ → leftSum array, rightSum array → manage a variable while travelling.

③ Optimised 2

Time - $O(n)$
Space - $O(1)$

Steps:

① Total sum of array, Tsum = 28

② Manage left sum in lsum variable, and decrease value from Sum,

	0	1	2	3	4	5	6	7
	2	1	4	2	3	4	10	7
Lsum =	0	2	3	7	9	12	16	
Tsum =	40	39	35	33	30	26	16	

Total sum = 42

8	9	10
2	4	3

-ve Number → To avoid Binary search

gndx = 6 → final Result

Steps:

① find total sum

② Travel on array, maye left sum and total sum

At ith gndx → include $(i-1)^{th}$ Element in left sum and Exclude i^{th} Element from total sum.

Find K Closest Elements 1 (Using Priority Queue)

Tuesday, 9 November 2021 9:59 PM

$X = 45$, $K = 3$

arr → 10 20 30 40 50 60

|arr[i] - x| → 35 25 15 5 5 15

Max – promity Queue.
on distance

30-15
10-35
20-25
30-15
40-5
50-5

[40, 30, 50] → Sort → [30, 40, 50]

Result