H.W :

M.E possible => 2 :

Moore's Voting algo

2 distinct

> N/3 :

3 distinct elements.

N	M.E 1	M.E. 2
20	c1	d1
17	c-1	d-1

Agenda : Why sorting?

Problems

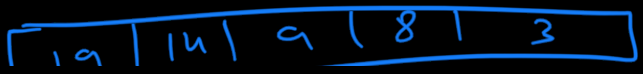
Comparators.

Sorting : Arranging ~~integers~~ data in ~~asc/desc~~ specific order based on a parameter.

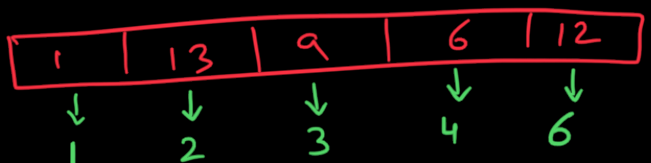
Buying on Amazing → sort by relevance / popularity / low to high price

sort a deck of cards
- ↳ By color
- ↳ By value
- ↳ Combination of value/type

Sorted in Asc order by value



Sorted in Desc order by value.



Sorted in Asc order by no. of factors.

Library sorting methods (T.C) $O(N \log N)$

S.C. : Depends ($O(1) \rightarrow O(N)$)

Sorting → Helps in searching

Q.1 Given an array of size N. You can remove one array element at a time.

Cost of removal = Sum of elements in the array just before removal.

Find min. cost to remove all elements.

A : [2, 1, 4]

remove 2 → 2 + 1 + 4 ⇒ 7

remove 1 → 1 + 4 ⇒ 5

remove 4 → 4 ⇒ 4

Total cost ⇒ 16.

Try diff order

remove 4 → 2 + 1 + 4 ⇒ 7

remove 2 → 2 + 1 ⇒ 3

remove 1 → 1 ⇒ 1

Total cost ⇒ 11

Quiz: 4, 6, 1

[4, 6, 1] remove 6 → 4 + 6 + 1 ⇒ 11

[4, 1] remove 4 → 4 + 1 ⇒ 5

[1] remove 1 → 1 ⇒ 1

Total ⇒ 17

Quiz 2: 3, 5, 1, -3 —sort in desc→ (5, 3, 1, -3)

Times.

5 → 1
3 → 2
1 → 3
-3 → 4

[3, 5, 1, -3] remove 5 → 3 + 5 + 1 + (-3) ⇒ 6

[3, 1, -3] remove 3 → 3 + 1 + (-3) ⇒ 1

[1, -3] remove 1 → 1 + (-3) ⇒ -2

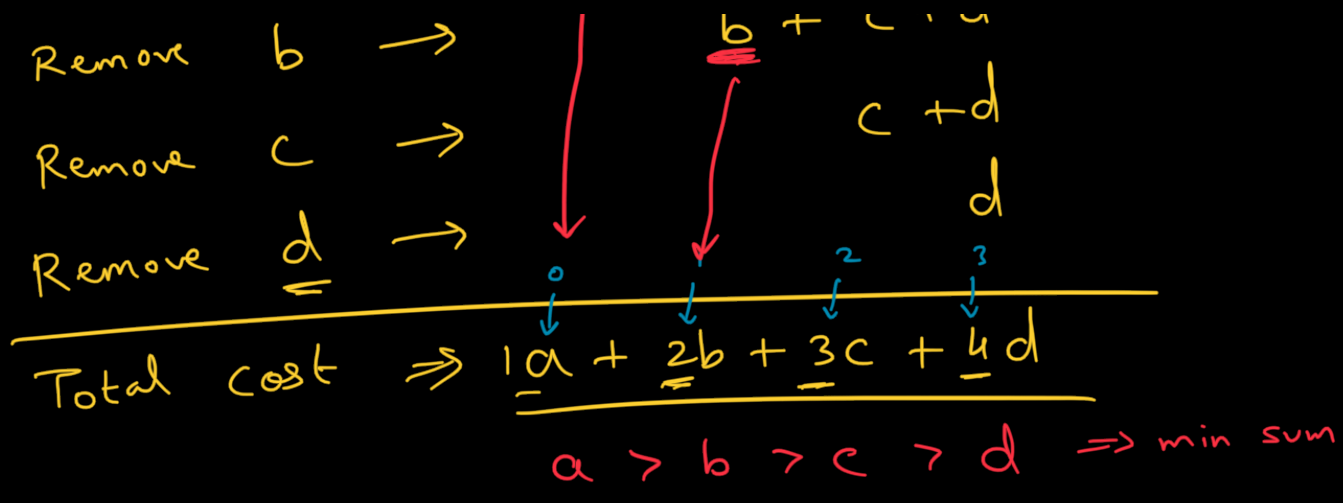
[-3] remove (-3) → -3 ⇒ -3

Total ⇒ 2

[a, b, c, d]

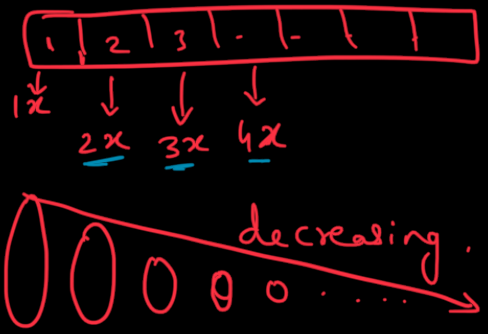
Remove a → a + b + c + d

T



Pseudo:

```
cost = 0
Sort (A)   (in desc order)
for(i=0 ; i<N ; i++)
{
    cost += A[i] * (i+1)
}
```



T.C ⇒ $O(N \log N)$ + $O(N)$ ⇒ $O(N \log N)$

$O(N \log N)$ — Sort

$O(N)$ — finding cost

→ Depends on sorting algorithm.

T.C ⇒ Depends on ... 0 0

Amazon Zeta **Q.2** Given N elements. Count no. of 'Nobel integers' present in the array. (Distinct elements)

Nobel integer: A[i] is a Nobel int

No. of element less than A[i] = A[i]

A : [1 , -5 , 3 , 5 , -10 , 4]

#less ⇒ 2 , 1 , 3 , 5 , 0 , 4

Ans = 3

Quiz [-3 , 0 , 2 , 5]

0 , 1 , 2 , 3

Ans = 1

Brute:

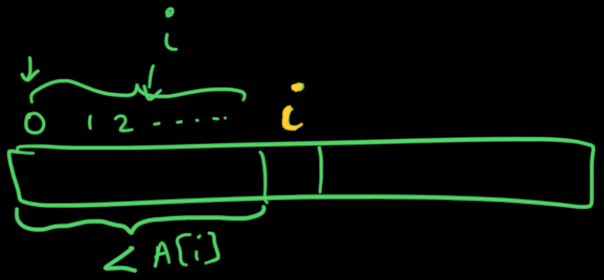
```
ans = 0
for ( i = 0 → N-1 )
```

T.C ⇒ $O(N^2)$

S.C ⇒ O(1)

```
{
    count = 0
    for( j = 0 → N-1)
    {
        if (A[j] < A[i])
            count++
    }
    if (count == A[i]) ans++;
}
ret ans;
```

Use Sorting



eg 1:

0	1	2	3	4	5
-10	-5	1	3	4	5

```
Sort (A)
for( i = 0 → N-1)
{
    if (A[i] == i)
        ans++;
}
```

T.C ⇒ O(N log N)

```
    return ans;
}
```

Q.2.b With duplicates.

Quiz : [-10, 1, 1, 3, 100]

	0	1	2	3	4
	-10	1	1	3	100
#less	0	1	1	3	4

Quiz : [-10, 1, 1, 2, 4, 4, 4, 8, 10]

	0	1	2	3	4	5	6	7	8
	-10	1	1	2	4	4	4	8	10
#less	0	1	1	3	4	4	4	7	8

Ans = 5

Quiz : [-3, 0, 2, 2, 5, 5, 5, 5, 8, 8, 10, 10, 10, 14]

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
	-3	0	2	2	5	5	5	5	8	8	10	10	10	14
			2	2	4	4	4	4	8	8	10	10	10	13

#les 0 1

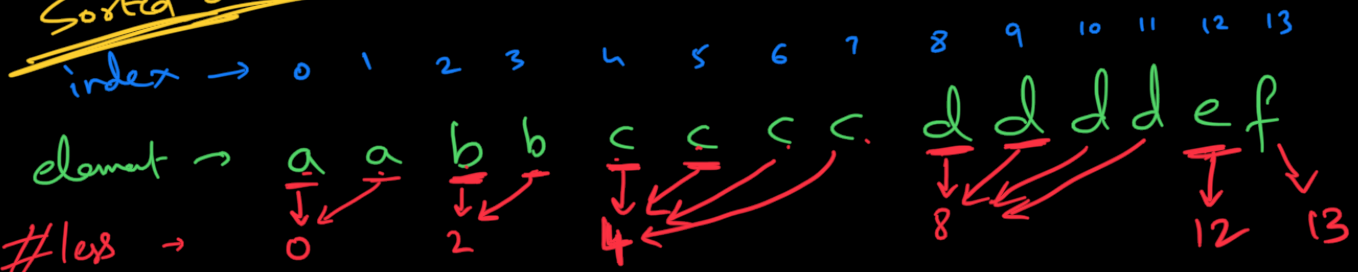
Ans = 7

Steps:

① Sort the array } O(NlogN)

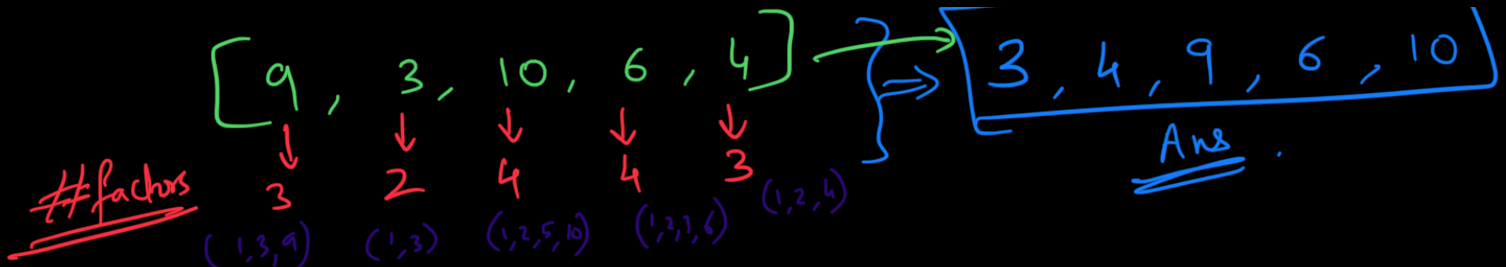
② } O(N)

Sorted data



Break 3 10:25

Q: Sort an array in ASC order by the no. of factors. If no. of factors is same, then sort based on val.



Comparator

Check if array is sorted

[1, 3, 5, 7, 9, 10]

a[i] < a[i+1]

[1, 4, 16, 9, 10]

```
for( i = 0 -> N-2)
    if (A[i] <= A[i+1])
        ; fine
    else break
```

1 4 16 3 10

At a time we only need

In any sorting algo → to compare 2 data points.

(Comparisor based)

Compare fn

↳ 2 argument (data-type that has to be compared)

↳ boolean

↳ 1/-1/0

↳ Based on parameters, returns which one is smaller & which one is larger (or come after the first element).

C++ : sort (v.begin(), v.end(), comp())

↳ int

↳ string

↳ obj.

Java : Arrays.sort(A, new Comparator<Integer>(){

.....}

)

[2, 3, 7, 1, 4] $\xrightarrow{\text{sort asc.}}$ {1, 2, 3, 4, 7}

```
boolean Comp( int a, int b)
{
    if(a<b)
        return true
    else ( )
        return false.
}
```

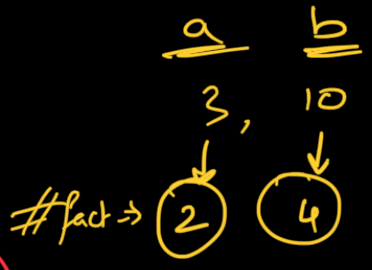
[2, 3, 7, 1, 4] $\xrightarrow{\text{sort in Desc}}$ [7, 4, 3, 2, 1]

```
boole  Comp(int a, int b)
{
    if(a > b)
        return true
```

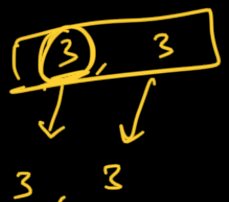
else return false.

}

a : [9, 3, 10, 6, 4]




```
bool comp(int a, int b)
{
    int fa = countfactor(a)
    int fb = countfactor(b)
    if (fa < fb)
        return true.
    else if (fa == fb)
        if (a < b)
            return true
    else return false
}
```



}

Custom comparator for sorting based on any parameter

Python : Asort (key = comp-to-key(cmp))

Google Amazon LinkedIn

Q. Given array of +ve integers. Return largest no. than can be formed using these integers.

A : [3, 30, 34, 5, 9]

3 30 34 5 9

3 4 5 3 0 9 3

largest → 9 5 34 3 30

long 10^{18}

18 18

Comp (int a, int b)

ab or ba

string ⇒ ab

ba ⇐ string

10^{18}

long: a b ab ba

25 16 1625

25 16

10 9 109 910

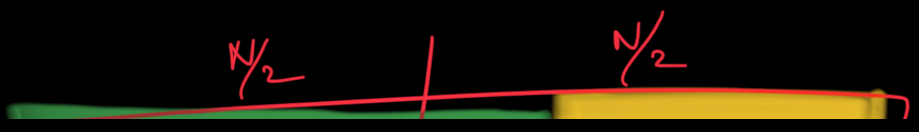
49 9 499 949

ab → 2516 ⇒ string

ba → 1625 ⇒ string

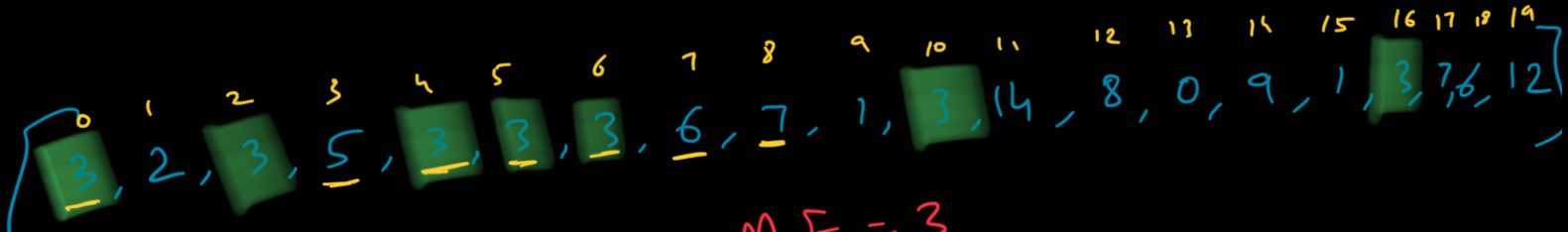
if(ab < ba)
return true
else

Doubts:



M.E → 1 element possible.

M.E → 2 elements possible



N = 20

$\frac{N}{3} = \frac{20}{3} = \boxed{6}$

M.E = 3

Cont = 7

distinct elements

(3, 2, 5)

(3, 6, 7)

3, 6, 7

[1, 1, 1, 2, 3, 5, 7] → M. Validate.

M.F = 7

M.F.1

M.F.2.