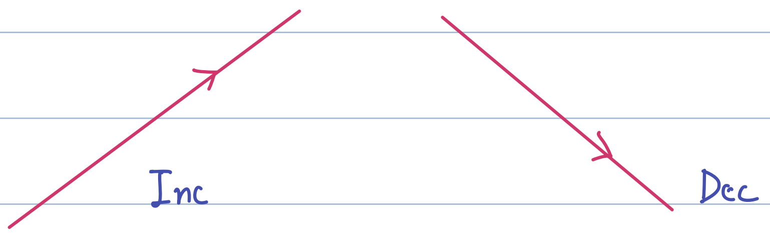
Friday : Contest 2

{ Bit Manipulation, sorting, modular arithmatic , string }

Sorting : Arranging data in icreasing or decreasing order based on a parameter

Cricketers	Salary	Runs
MSD	1Cr ③	7000
Virat	5Cr ④	10000
SKY	0.5Cr ①	3000
Jadeja	1Cr ②	2500



{ 3 5 7 9 10 }   Inc

based on value

{ 1 , 13, 9 , 6 , 12 }   Inc

factors: 1 2 3 4 6   based on factors

Algorithms

1)	A1	Merge sort	Advanced
2)	A2	Quick sort	
⋮		Selection sort	
		⋮	

· Problem solving based on sorting

How to sort an array in your language?

	Python	C++	Java
l =	l.sort()	sort(l, l+n)	Collection.sort(l)

l.sort() $\downarrow$ Inc

$$\boxed{\begin{array}{l} TC: O(N\log N) \\ SC: O(1) \end{array}}$$

Zomato

Q1) Given an array of N elements. Remove all elements from array one by one.

Cost of removing any element :

Sum of all array elements at that time (before deletion)

Return minimum cost of deleting all elements.

$$\begin{array}{cccc} & 0 & 1 & 2 \\ A: & \cancel{2} & \cancel{1} & 4 \end{array}$$

Deleted 1: $2+1+4 = 7$

Delete 2: $2+4 = 6$

Delete 4: $4 = 4$

$$\begin{array}{cccc} & 0 & 1 & 2 \\ A: & 2 & 1 & 4 \end{array}$$

Total cost : 17

$\{\cancel{4}, \cancel{6}, 1\}$

Del 6: $6+4+1 = 11$

Del 4: $4+1 = 5$

Del 1: $1 = 1$

$\boxed{TC: 17}$

Remove the max element.

$\downarrow$

2nd largest $\longrightarrow$ 3rd largest $\overset{\cdots}{\longrightarrow}$ Smallest

Why? 70

$\boxed{\{ a_0^{\propto} \quad a_1^{\propto} \quad a_2 \quad a_3^{\propto} \}}$ 4

Del a_0 : $a_0 + a_1 + a_2 + a_3$

Del a_1 : $a_1 + a_2 + a_3$

Del a_3 : $a_2 + a_3$

Del a_2 : a_2

✓

$\boxed{a_0 + 2a_1 + 4a_2 + 3a_3}$ Minimize

✓

Higest coeficient needs to be multiplied by least value

$\boxed{a_2 < a_3 < a_1 < a_0}$

Obs: Del elements in decreasing order

1st lowest element × N

2nd lowest × N-1

⋮

largest × 1

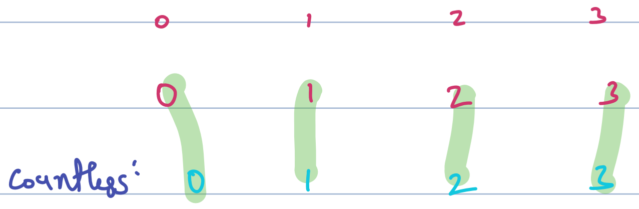
```
l.sort()        // asc order

mincost = 0
temp = N
for (i=0 ; i<N; i++) {
    mincost += temp * A[i]
    temp --
}

return mincost
```

TC : $O(N\log N + \cancel{N})$

SC: $O(1)$



Q2) Nobel integer

unique elements

Given an array of size N having distinct elements. Count number of nobel integers.

Nobel integer: Any element A[i] for which
count of elements less than it = A[i]

A:	0	1	2	3	4	5
	-1	-5	3	5	-10	4
Countless	2	1	3	5	0	4

$\boxed{\text{Nobel : } 3}$

	0	1	2	3	4	5	6	7
A:	4	8	3	2	-1	1	7	6

{ -3 0 2 5 }

countless 0 1 2 3

$\boxed{\text{nobel} = 1}$

Brute force: For every index i
count no. of eles < A[i]
count == A[i]
# nobel

TC: O(N²)

Optimised approach:

0	1	2	3	4	5	6	7
4	8	3	2	-1	1	7	6

sort

	0	1	2	3	4	5	6	7
	-1	1	2	3	4	6	7	8
countless:	0	1	2	3	4	5	6	7

$\longrightarrow$

$\boxed{\text{countless} = i}$

```
l.sort()    nobel = 0                    TC: O(NlogN)
for(i=0 ; i< N ; i++){
|       if (A[i] == i){
|       |   nobel++
|       }
}
```

Data can Repeat { Nobel integers }

Ex

	0	1	2	3	4	5
Ex	{ 0	2	2	3	3	6 }

	0	1	2	3	4	5	6	7	8
Ex :	{ -10	1	1	1	4	4	4	7	10 }

	0	1	2	3	4	5	6	7	8
Ex:	{ -10	1	1	2	4	4	4	8	10 }
countless :	0	1	1	3	4	4	4	7	8

Ans : 5

	0	1	2	3	4	
	{ -10	1	1	3	100 }	output : 3
countless :	0	1	1	3	4	

	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 }
countless :	0	1	2	2	4	4	4	4	8	8	10	10	10	13

Ans : 7

1) Brute force remains same

2) $\boxed{\text{countless} = i}$ $\propto$

✓ obs 1) Countless remains same for equal values

✓ 2) First occurances countless = i

A[i] is repeating or first occurance?

→ check previous value

```
l.sort()
nobel = 0          ,    countless = 0
for(i=0 ; i < N ; i++){
    if (A[i] != A[i-1]) {
        countless = i
    }
    if (countless == A[i]){
        nobel++
    }
}
```

TODO: fix this code

$\boxed{10:39 - 10:50}$

Q Given an array, sort the array based on number of factors?

0	1	2	3	4	5	6
{ 9	3	4	8	16	37	6 }