Contest - 1.

9:05.

1. No of elements to remove to make B the largest.
2. Vowels in a range
3. Generate A 1's followed by B 0's.
4. Matrix operations.

Keerthi.
CS-I (SDE-3) Adobe, flipkart, wipro.
7 years
5 years into teaching.

1. No of elements to remove to make B the largest.

i/p: A = [2, 4, 3, 1, 5].
B = [3].

o/p : ans = 2.

i/p : A = [1, 4, 2]
B = [3].

o/p : ans = -1.

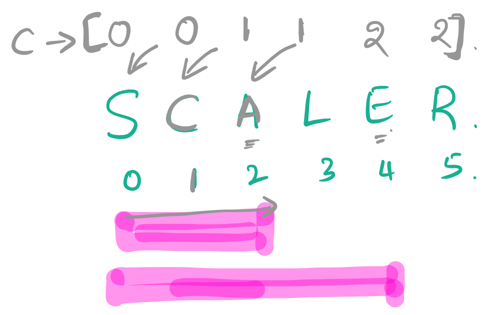
```
int solve(int[] A, int B)
{
        boolean isBPresent = false;
        int elementsToRemove = 0;
        for(int i=0; i< A.length; i++)
        {
              if (A[i]==B)
                    isBPresent = true;
              if ( A[i]>B)
                    elementsToRemove++;
        }
        return   isBPresent? elementsToRemove : -1; }  ternary operator.
    }
```

2) Vowels in a range.

i/p : string , A = "scaler"

B = [[0,2]
      [2,4]]

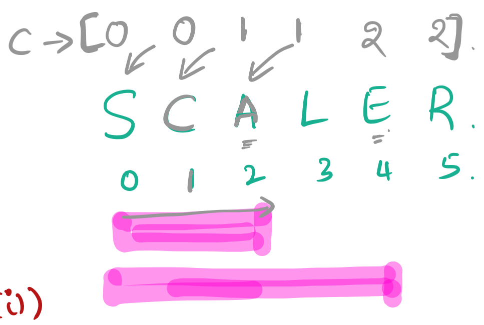
o/p: [1 ✓
       2] ✓



Brute force.

[0 2] -> — output.
[2 4] —

```
ArrayList<Integer>  solve ( String A,  ArrayList<ArrayList<Integer>> B)
{
        ArrayList<Integer> outputList = new ArrayList<>();
        int[]  vowelsTillI = new  int[A.length];
        if (isCharacterAVowel(A.charAt (0))
        {
              vowelsTillI[0] = 1;
        }
        for (int i=1; i<A.length; i++)
        {
            if (isCharacterAVowel(A.charAt (i))
            {
                  vowelsTillI[i] =  1+ vowelsTillI[i-1];
            }
            else
            {
                  vowelsTillI[i] =     vowelsTillI[i-1];
            }
        }
```



```
for( ArrayList<Integer> list : B)
{
        int sum = vowelsTillI[list.get(1)]
                    - vowelsTillI[list.get(0)];
        if (isCharacterAnVowel(A.charAt(list.get(0)))
        {
            sum++;
        }
        outputList.add(sum);

}
return outputList;

}
```

$B = \begin{bmatrix} [0,2] \\ [2,4] \end{bmatrix}$

length of string.
↑
TC: O(K)+
SC: O(K)

$[\underline{2 \quad 10}] \rightarrow [10-2+1]$

$[b-a+1]$

3) Generate A 1's followed by B 0's.

```
int solve(int A, int B)
{
    int ans=0;
    for(int i = B ; i<A+B ; i++)
    {
        int lastBit = 1<<i ;
        ans = ans| lastBit;
    }
    return ans;
}
```

A 1's.
4

$\boxed{1000}$
↑
$\boxed{16}$.

$[1<<5]$

100000

→[0 0 0 0 0 0 0 0 0]
1 0 0 0
0 0 0 0 1 0 0 0

A = 10, B = 4.
1 0 0 0 0

$\boxed{1}$ → 10 times.
0 0 0 1 1 0 0 0

0 0 0

1 $\boxed{1}$ 1

1 1 1 1 1 1 1 1 1 1 0 0 0 0

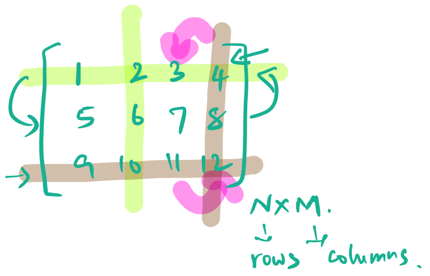
$\boxed{1}$ → 4 times
0 0 0 0 0 0 0 0 0
1 0 0 0 0 +
1 0 0 0 0 0 +
1 0 0 0 0 0 0 +

1 1 1 0 0 0

$\boxed{10:20}$

4. Matrix operations.

$N=3.$
$M=4.$



i/p: N, M,

Q queries.

(i) C_1 C_2 : swap ~~cl~~ columns C_1 & C_2.

(ii) R_1 R_2 : swap rows R_1 & R_2.

(iii) X1 Y1 X2 Y2 : OR of values

(iv) X1 Y1 X2 Y2 : AND of values.

[0 1] [2 3].

$\textcircled{2}$ $\textcircled{12}$

```
 0010
 1100
 ----
 1110 = 14.
```

Brute force solⁿ.

store the entire matrix & do the operations.

$m=4, n=3.$

[0, 2]. = 3 ✓
[1, 3] = 8 ✓
[2, 1]. = 10. ✓

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

$$\boxed{[\text{row}][\text{col}] = 1 + \text{col} + (\text{row}) * m.}$$

0	2	= 1 + 2 + 0 * 4	= 3
1	3	= 1 + 3 + 1 * 4	= 8.
2	1	= 1 + 1 + 2 * 4	= 10.

(0, 0) = 1.
(0, 1) = 2.
(0, 2) = 3.

$N=10,\ M=8.$

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\ & & & & & & & \\ & & & & & & & \end{bmatrix}$$

$$\begin{array}{c|cccc} & 0 & 1 & \cancel{2} & 3. \\ \hline 0 & 1 & 2 & 3 & 4 \\ 1 & 5 & 6 & 7 & 8 \\ 2 & 9 & 10 & 11 & 12 \end{array}$$

(c_1, c_2)

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 1 & 3 & 2 & 4 \\ 1 & 5 & 7 & 6 & 8 \\ 2 & 9 & 11 & 10 & 12 \end{array}$$

0 → 0.
1 → 1
2 → 2
⋮

c1 → 1, 2.
c2 → 2, 1

A [1][1] = 6 , before swapping.
A [1][2] = 6. ,

R0 → 0
R1 → 1.
R2 → 2
R3 → 3 → 4
R4 → 4 → 3.
RK → K.

C0 → 0
C1 → 1 } C1 → 2.
C2 → 2 } C2 → 1

CK → K.

C1 & C2.
R3 & R4

```
main( String[] args)
{
        Scanner sc = new Scanner(System.in);
        // read n, m, q.
        long[] r = new int[100005];
        long[] c = new int[100005];
        for(int i=0; i<n; i++)
        {       r[i]=i;
        }
        for (int i=0; i<m; i++)
        {       c[i]=i;
        }
        for(int i=0; i<q; i++)
        {       int t = sc.nextInt();
                if(t==1) {
                        int c1 = sc.nextInt();
                        int c2 = sc.nextInt();
                        long temp = c[c1-1];
                        c[c1-1] = c[c2-1];
                        c[c2-1] = temp;
                }
```

q=3.

2 3 4
1 2 3
3 1 2 2 2

1 ← C1 , C2 → 2.

C1 = 1 } C1 = 2
C2 = 2 } C2 = 1.

C1 = 10 ✓ →
C2 = 20 ✓
5.

~~C~~

[0 1 2 3 4 5]
0 1 2 3 4 5.

c[c1-1]

```
if(t==2) {
    int c1 = sc.nextInt();
    int c2 = sc.nextInt();
    long temp = r[c1-1];
    r[c1-1] = r[c2-1];
    r[c2-1] = temp;
}
if (t==3)
{
        read  x1, y1, x2, y2;
        long a= 1+c[y1-1]+r[x1-1]*m;
        long b= 1+c[y2-1]+r[x2-1]*m;
        print (a|b);

}
if (t==4)
{
        read  x1, y1, x2, y2;
        long a= 1+c[y1-1]+r[x1-1]*m;
        long b= 1+c[y2-1]+r[x2-1]*m;
        print (a&b);

}
        }
    }
}
```

A[x2][y2] ← b.

a ↓ $\boxed{A[x1][y1]}$.

= ~~**~~ 1+$\underline{\underline{y1}}$+$\underline{\underline{x1}}$*m.

9 queries.

2 2 3
1 1 2
2 1 1 2 1.

A[row][col] = [1+ col + row * m]

~~RO~~
~~R~~

$[0 \quad \boxed{1}^{2} \quad \boxed{2}^{1}]$

R 0 1 2

$C \; [0 \quad 1 \quad \boxed{2}^{3} \quad \boxed{3}^{2}]$

0 1 2 3

A[1][1] = 1+ col[1] + row[1] * m.
= 1+ 1 + 2 * 4 = 10.

A[2][1] = 1+ Col[1] + row[2] * m
= 1+ 1 + 1 * 4 = 6.

$m=4, \; n=3.$

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

C, [2 2 3]

$$C, \begin{bmatrix} 1 & 2 & 4 & 3 \\ 5 & 6 & 8 & 7 \\ 9 & 10 & 12 & 11 \end{bmatrix}.$$

[1 1 2]

$$C, \begin{bmatrix} 1 & 2 & 4 & 3 \\ 9 & 10 & 12 & 11 \\ 5 & 6 & 8 & 7 \end{bmatrix}.$$

(10) 4 (6).