PAGE :
DATE : / /

~~CHEFX~~ Q using sqrt decomposition:

Let us consider an array (1-indexed) of 'n' elements.

$a_1, a_2, a_3 \ldots a_n$.

So, what we will do is, we divide the ~~element~~ array into blocks. Each block will contain $\lceil sqrt(n) \rceil$ elements in it.

In total, how many blocks will be there?

i.e. $\lceil sqrt(n) \rceil$.

So we maintain a 2D array to store this information.

Let us call this array as BLOCK[][] {array of sx * sx}.

The array contains elements in block wise manner.

i.e. Let $\lceil sqrt(n) \rceil$ = sx.

block 1 will contain elements:

$a_1 \;\; a_2 \; \ldots \; a_{sx}$.

block 2 will contain elements:

$a_{sx+1}, a_{sx+2} \ldots a_{sx*2}$.

⋮

So on till block sx.

It is not necessary that the last block contains sx elements.

No. of elements in it may range ~~in~~ [1 to sx].

Signature

We will maintain a map ~~and~~ for each block and an array containing cumulative XOR values of elements of each block.

First talk the array. Let us call it XVAL[]

It will contain ~~the~~ sz elements in it.

And XVAL[i] represents the cumulative XOR value of the elements of i^{th} block.

Let us ~~know~~ now know the rest the map.

We will have a map for each block.

So, let's take an array of maps of size sz.

i.e. map < d_t, d_t > name[];

Now what will map store?

for a block, a map will contain count of the step wise XOR value.

What does it mean?

~~i.e.~~ Consider the block:

2, 3, 4, 5.

So, the map for it will contain ~~cont~~ count for XOR values:

2, 2^3, 2^3^4 & 2^3^4^5.

Similarly for other blocks:

i.e. map< ll, ll > M[SZ]

for some XOR value XOR_val in i^{th} block:

++ map[i][XOR_val];

Signature

Scanned by CamScanner

DATE : / /

This was the whole build part.

Now, let us know how to update the above data we have stored when an update query is encountered.

Consider we have to update value in array at ith position. Updating that will ~~the~~ affect the BLOCK[i][] and map for the ~~th~~ block in which it lies.

If we are given position & new value for update we can update the array in O(1).

Now, let us update the 2D array BLOCK and the map.

For that we have to find out the block in which it lies.

Let us find out block no. :

if (position % sz == 0)
    then block no. = position/sz;
else
    block no. = (1 + position/sz);

Let us take an example :

ar[] = {1, 2, 3, 5, 4, 7, 8, 9, 6} (1-st indexed array)

There are 9 elements in the array.

block size, sz = $\lceil sqrt(9) \rceil = 3$.

Signature

Let us find the ~~pos~~ the block no. for position 4 & 6 :-

$4 \% 3 == 1$.

So $\lfloor 4/3 \rfloor = 1$ as per condition ;

block no := 2.

$6 \% 3 == 0$.

$\lfloor 6/3 \rfloor = 2$ block no. := 2.

if we group the index of the array, we will find that 4 & 6 lies in 2nd block.

Now, we have to ~~have~~ find in which position ~~does~~ the element is to be update in the block no. we got.

So, consider position 5.

Block no. = 2.

As array are 1-indexed.

and block no. 2 contain 4, 5, 6 :

So, 5th position ~~elements~~ ~~cons~~ corresponds to

Block [2][2].

i.e ~~ith~~ for any ith element in the array:

block no. is i/sz if $i \% sz == 0$

or $1 + i/sz$ if $i \% sz != 0$.

To get the secondindex for block:

$i - ((\text{block no.} - 1) * sz)$;

PAGE :
DATE : / /

Then update BLOCK with the new value.

Introduction of this new value also effect the XOR values of the block.

So, we have calculate again the cumulative XOR value and store it in corresponding position in XVAL.

And we have first clear the map for the block and re-do the counting as previously for the new values.

Now comes the query part (finding the count for 'k')

few properties of XOR:

$a \wedge a = 0$ $\quad a \wedge b = c$ $\quad c \wedge b = a$.
$a \wedge 0 = a$ $\quad c \wedge a = b$

i.e. $a \wedge b \wedge b = ?$

$\boxed{a \wedge (b \wedge b) = a \wedge 0 = a}$.

for finding the count for any 'k' in range [1 to i]
'i' be any index within range of array.

Find out the block no. for the 'i' index.

Now, using the above property:

Consider this example:

Signature

/ /

Let you be given i^{th} position & ~~cot~~ 'k'.

Let the block no. be 'n'.

Then first traverse n-1 block.

for 1 to n-1 blocks:

for 1^{st} block, check its map if k' is there or not.

If there add the count of it into your final ans.

Tricky.

Then go to next block and find for k^XVAL[1].

Why k^ $\underbrace{XVAL[1]}_{\text{cumulative XOR for block 1}}$

See k^b^$b = k$. Okay?

and a^$a = 0$.

If we find k^XVAL[1] in the map of the 2^{nd} loop

that means, we were able to find k.

PAGE :
DATE : / /

For last block check each element by computing XOR -
cumulative XOR of n-1 blocks = temp.
temp ^ = each element of last block.