Educational Codeforces 126

Solutions now for A and B with explanation

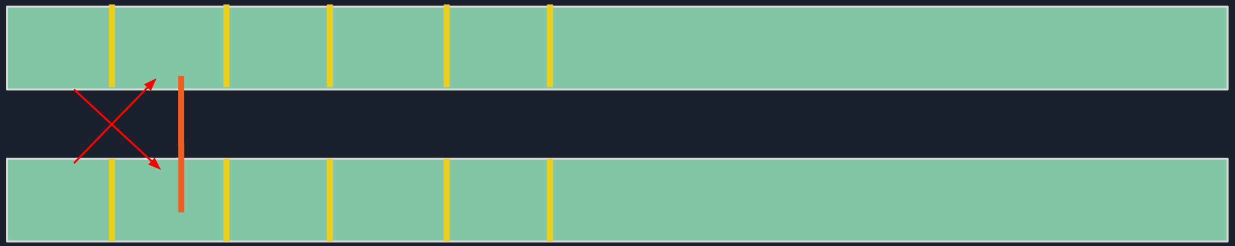
Problem A

Given : https://codeforces.com/contest/1661/problem/A refer the link for the given statements

As the swaps can be done on same position of the 2 arrays a and b we should get the minimum value on each iteration for |ai-ai+1|+|bi-bi+1| and here what we can do is we should make that value as minimum as possible and only way we can do is make the sum mentioned above to be as minimum as possible in each step.

So this can be done by iteration through each element and check |ai-ai+1|+|bi-bi+1| is minimum or |ai-bi+1|+|bi-ai+1|. If the later is minum then swap ai+1 and bi+1 else dont do anything and by doing this we get the minimum sum on each iteration across the array

Compare the adjacent positions as while iteration across the array and get the minimum each time



The red arrow marks denotes the comparison to be done each time and the orange denotes the swap condition on each iteration which means swap the elements when we get the minimum sum by comparison we swap everytime

Problem B

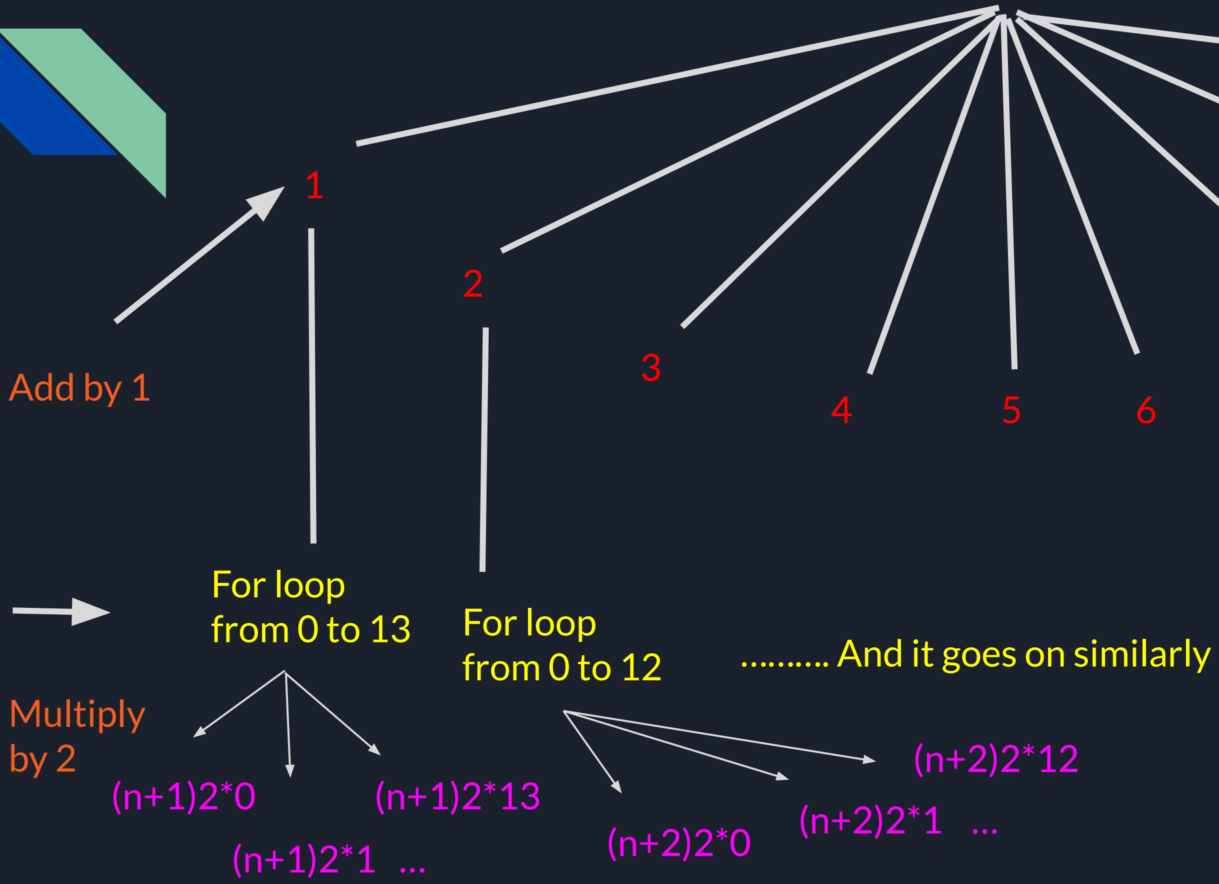
Given : https://codeforces.com/contest/1661/problem/B refer the link for the given statements

See if u guys clearly notice 32768 = 2^15 soany number which is a multiple of 2^15 will get accepted and for that we have to multiply it 15 times.

So the boundary value runs upto multiplying the number upto 15 times with 2 to be divisible by 2^15 but here we can use bruteforce method to solve this problem as we analyse each possible method to get the number a divisor of 2^15. What we gonna do is we check each case from adding one 0 times and multiplying by 2 0 times to adding 1 14 times and multiplying by two 14 times and each time check if its a divisor of 2^15 and update the counter variable.

Note : The total steps should be less than 15 so when we add by 1 we can multiply only 13 times and similarly reduces in the above manner

Tree



Add by 1

Multiply by 2

For loop from 0 to 13

(n+1)2*0

(n+1)2*1 ...

(n+1)2*13

For loop from 0 to 12

(n+2)2*0

(n+2)2*1 ...

(n+2)2*12

.... And it goes on similarly

We get the required solution by updating the counter variable everytime until the loop breaks which is min(15,counter_variable) and if the counter variable >15 then break