Epiphany 11.0 Editorial

1. Roll it

Each character can be rolled back to starting character that is 'a' if it exceeds the value of z.

Since there are only 26 characters, We can take modulo 26 to roll back. This way we can change each character in the string and print

the final string.

2. Alice and Bob

The problem is based on observation.

Firstly the sum is fixed that means no matter how many times we perform operation it will be constant.

We can observe that if x and y are the two values:

after p operations x will be:

x = (x*(2^p))%(x+y) (since we are doubling the values and we can always bring the values lesser than the sum if it ever exceed it)

y = sum - x (since sum is constant)

3. Max. Xor Pair

The max xor pair is when the elements differ in the most significant bit, as the array is sorted this can be done using binary search.

First find the most significant bit of the last element this can be done in 2 operations, then do binary search by checking the middle element (2 operations to find each mid element. 1 to mid, and 1 to mid-1).

4. Can you beat the dragon?

This is a greedy problem divide the arrays into 2 parts based on boost whether it is positive or negative, then sort the positive array in increasing order of difficulty, and the negative one in decreasing order of difficulty. Append the negative array to the positive one and then iterate through the tasks in that order if health goes negative or difficulty is greater than health then it is not possible otherwise it is.

5. Can you still beat the dragon?

For the positive boost tasks all of them can be included, to select the negative boost tasks you will have to find that using Dynamic programming after sorting them by decreasing difficulty and can be done O(n^2) time.

6. Switch Tree

If you enumerate each node using pre/post-order dfs you can have a range of values l to r which denoted the nodes in its sub tree. Now for operation 1 it is setting bits of nodes in range l to r of node k and for operation 2 we have to unset the bit of the node. For query type 3 if there is any unset bit in range l to r the answer will be 0 otherwise 1. To do these operations quickly a segment tree can be used.

There are other ways to solve this problem as well.

7. Sheong Gi Hun

If we use a naïve approach using three parameters N, M, K. We can see that it is an overlapping sub problem but if we do memorization on this it will give us TLE and MLE. So we have to think something else. Here you can note that the order of elements doesn't matter and we have to make sure that there or all elements should be K. So we can use bit manipulation. Here if ith bit is active (0 <= I <= 15) in K then we have n choices in which we can activate that bit until and unless M > 0.

8. Cube game

If we find for each cube the cube on which it will jump then half problem is solved then we can easily use binary lifting and solve each query in O(logK) time complexity. But here the catch is to find next cube for each cube. If we use a naïve approach by iterating over all combination it will take us O(N ^ 2) time. We can optimize it further using segment tree with lazy propagation. Initially create 1 as root node. Then calculate distance from 1 to all other node. When again do dfs and simultaneously find next cube.

So overall time complexity for this solution will be O(NlogN + QlogK)