Tutorial for CF862D

有一个长度为 n（$2 \leq n \leq 1000$） 的 01 串（未知），保证至少有一个 1 和一个 0，你可以构造任意的字符串，然后人机会交互地（最多15次）回答该串与原串的编辑距离（该串与原串不同位置的个数）. 求原串的任意一个 0 下标 和 1 下标.

1.考虑询问一个全 0 串，得到的编辑距离 c_1 是 原串中 1 的个数

2.我们容易想到二分查找下标是相邻的 01 的区域，开始我们是区间 $[1,n]$，设二分位置为 mid, 我们让 mid之前全为 0，让 mid 之后全为 1,得到编辑距离 c_2，设$len = n - mid$.$[1,mid]$的1的个数是x，$[mid+1,r]$的1个数是y
这样对于c_2, 在 mid 之前对它的贡献就是x，在 mid 之后对它的贡献$len - y$

$$\begin{cases} x + (len) - y = c_2 \\ x + y = c_1 \end{cases}$$

$x = \frac{c_1+c_2-len}{2}, y = c_1 - x$

3.推广到一般情况，我们现在得到一个区间 $[l,r]$，维护$[1,l-1]$中 1 的个数是$sumL$, $[r+1,n]$中 1 的个数是$sumR$,于是得到
$x = \frac{c_1+c_2-len}{2} - sumL,$
$y = c1 - (x + sumL) - sumR$

4.最后我们只需要在二分的时候判断应该往哪边二分，同时根据哪个区域是全1，哪个区域是全0，来记录0和1的下标.由于至少包含一个01区域，所以不存在二分无解的情况.
时间复杂度 $log(n)$.空间复杂度$O(1)$.

```
int l = 1, r = n;
int c1 = query(make(n));
int pos0 = -1, pos1 = -1;
int sumL = 0, sumR = 0;
while (l < r){
    int mid = l+r >> 1;
    int c2 = query(make(mid));
    int len = n - mid;
    int x = (c1+c2-len)/2-sumL;
    int y = c1-(x+sumL)-sumR;
    int L1 = mid-l+1, L2 = r-mid;
    assert(x >= 0);
    assert(y >= 0);
    assert((c1+c2-len)%2==0);
    if (y == 0) pos0 = mid+1;
    if (y == L2) pos1 = mid+1;
    if (x == 0) pos0 = mid;
    if (x == L1) pos1 = mid;

    if (x > 0 && L1 != x){
        sumR += y;
        r = mid;
    } else if (y > 0 && L2 != y){
        sumL += x;
        l = mid+1;
    } else if (x == 0){
        l = mid+1;
    } else if (x == L1){
        sumL += x;
        l = mid+1;
    } else if (y == 0){
        r = mid;
    } else {
        sumR += y;
        r = mid;
    }
}
```