Build The Grid

> Given is a square grid of $N \times N$ squares. Your task is to paint each square of the grid either white or black such that:
>
> - The white squares are connected: for any two white squares, you can go from one to the other by moving only between white squares that share a side.
> - Each black square shares a side with at least one white square.
> - Denote the number of black cells in the i-th row as p_i. The sequence $P = (p_1, p_2, \ldots, p_N)$ is then a permutation of integers between 0 and $N-1$, inclusive.
> - Denote the number of black cells in the j-th column as q_j. The sequence $Q = (q_1, q_2, \ldots, q_N)$ is then a permutation of integers between 0 and $N-1$, inclusive.
>
> It can be shown that such a construction always exists.

我们考虑 $N-1$ 的那行和那列：

```
  W B
W W W W W
B W B B B
  W B
  W B
```

那么我们发现右下的那个东西就不连通了。如果这样：

```
      B
      B
      B
B B B B
```

那右下的 `B` 就完蛋了。那么我们这样：

```
    W B
    W B
W W W W
B B W B
```

继续这样向上，就好了。

Juggler's Trick

> N balls are lined up in a row from left to right. Each ball may be either uncolored (white), blue, or red. Additionally, two integers r and b are given. Let us represent the ordering as a string consisting of letters 'W', 'B' and 'R' for uncolored, black, and red balls, respectively.
>
> For each trick, the juggler may choose a *combo* of $r+b$ consecutive balls such that there are exactly r red balls and exactly b blue balls, in any order, and remove them. The remaining balls are concatenated while keeping their relative order. For example, if the initial order was "RRBRBBR", and the juggler removed "RBB", the result would be "RRBR".
>
> Before the process starts, the juggler shall paint each uncolored ball either red or blue. The juggler wants to do as many tricks as possible. Find the maximal number of tricks if the juggler will choose the colors for the uncolored balls optimally.
>
> **Input**
>
> The first line of input contains three integers N, r and b ($1 \le N \le 2 \cdot 10^5$, $1 \le r, b \le N-1$, $r+b \le N$): the total number of balls, the number of red balls in a combo and the number of the blue balls in a combo, respectively. The second line contains the string S. This string encodes the initial order of balls and consists of exactly N letters 'B', 'R' and 'W', representing blue, red, and uncolored balls, respectively.

我们首先发现选取的是一些连续段，连续段内部可能纵横交错。一个长度为 $k(r+b)$ 的连续段是合法的，一个必要条件是 $R=kr, B=kb$；事实上，这也是充分的。

我们直接从左向右 dp，设 f_i 表示前 i 个的最大次数，转移方程：

$$f_i = \max_{j<i}\left\{ f_j + (i-j)\cdot\frac{1}{r+b} \mid SR_i - SR_j \le (i-j)\cdot\frac{r}{r+b}, SB_i - SB_j \le (i-j)\cdot\frac{b}{r+b} \right\} \tag{1}$$

用 $r+b$ 个二维线段树就可以维护。

中位数

> 给定一个长度为 n 的整数序列 a，你可以进行以下操作不超过 k 次：
>
> - 选择一个区间 $[l,r]$ 满足 $1 \le l \le r \le n$，并将 $[l,r]$ 中的所有数替换为这个区间的中位数。
>
> 你要使得操作后 a 的**最小值最大**。
>
> 关于此处中位数的定义：对于一个长度为 len 的序列，其中位数定义为该序列中第 $\lceil \frac{len}{2} \rceil$ 小的数。
> $n \le 4 \times 10^5$。

二分后就变为只有 `01` 序列的问题。然后我们发现是将某个区间变成 1，然后每次都在这个区间的基础上把区间扩大。比如 $[5,9] \to [3,10] \to [1,10]$。

观察 1：$k \le \log n$。因为区间的长度是倍增的。

观察 2：有用的区间只有 $O(n)$ 个，对于每个 l 维护 $f_l = r$ 表示它能延伸到的最靠右的右端点。

观察 3：设 S_i 表示视 0 为 -1 时的前缀和，那么只有 S_{l-1} 的前缀最小值上的 l 有用，只有 S_r 的后缀最大值上的 r 有用。

我们考虑不是对 $[l,r]$ 枚举 $[L,R], L \le l, r \le R$，而是对于每个 L 找到一个 $[l,r]$ 使得 R 最大。而一个 $[l,r]$ 的作用便是使 S_{L-1} "减少" $d=(r-l+1)-(S_r-S_{l-1})$；能匹配的 (R,S_R) 就必须两维都大于 $(r, S_{l-1}-d)$。

我们从右到左维护可以有的区间 $\{[l,r]\}$，那么 (r,d) 肯定构成一个后单调栈；每次看 r 最大但是 d 也最大的区间，如果能转移，它当然最优；如果不能转移，因为 S_{l-1} 自右向左递增，所以它之后都没用了，所以弹掉。

总复杂度 $\mathcal{O}(n\log^2 n)$