高精度（c++需要用）

A+B 两个大整数相加 → a和b的位数10^6

A-B 两个大整数相减 --- 10^6

A*a 一个大整数乘一个小整数

↓ ↘

len(A) $\in 10^6$ a用long long

A÷a 一个大整数和一个小整数相除

① 大整数存储

一个九位整数 → 会把每个存进数组里、

↑ ↘

1 2 3 4 5 6 7 8 9 数组中存的是

9 8 7 6 5 4 3 2 1

一个位存进数组第1位，因为 ↵

要考虑进位，高位可能需要补位

②

$$\begin{array}{r} A_3\ A_2\ A_1\ A_0 \\ +\quad B_2\ B_1\ B_0 \\ \hline C_2\ C_1\ C_0 \end{array}$$

A、B为两个数组

$A_i + B_i +$ ⊕

↓

进位

用数组模拟加法过程

减法：

$A_3\ A_2\ A_1\ A_0$

$-\quad B_2\ B_1\ B_0$

借位 ↑

$A_i - B_i - t$ ⟨ ≥ 0 $A_i - B_i - t$；< 0 $A_i - B_i + 10 - t$

做 A - B 的时候 分两步

① $A \geq B$，→ $A - B$

② $A < B$，→ $-(B - A)$

t ⟨ ≥ 0 t；< 0 $t + 10$ → t 可能有两种情况，可以直接这样写 $(t + 10)\%10$

乘法：

A_5 A_4 A_3 A_2 A_1 A_0

× $\left[\frac{A_0 \times b}{10}\right]$

?

$A_0 \times b \% 10$

$(A_1 \times b + t) \% 10$

进位

A 1 2 3

B 1 2

C_3 C_2 C_1 C_0

$C_0 = 3 \times 12 \% 10 = 6$

$t_1 = 3 \times 12 / 10 = 3$

$C_1 = (2 \times 12 + t_1) \% 10 = 7$

$t_2 = 2$

把B看成一个整体和A乘

$C_3 = (1 \times 12 + t_2) \% 10 = 4$

$t_3 = 1$

$C_3 = 1$

除法：

```
      0 1 1 2
    ---------
11 | 1 2 3 4
     0
    ---
     1 2
     1 1
    -----
       1 3
       1 1
      -----
         2 4
         2 2
        -----
           2
```

除法是从高位开始计算

前缀和：

$a_1, a_2, a_3 \cdots a_n$ 长度为n的数组。

前缀和数组：$S_i = a_1 + a_2 + \cdots a_i$

↓

原始数组前i项和

处理边界

令 $S_0 = 0$

① 如何求 S_i ⇒ `for(i=1; i<=n; i++)`

`S[i] = S[i-1] + a[i];`

② 前缀和的作用

快速求出原数组中一段数的和。

eg：求原数组中 $[l, r]$ 的值。

如果没有前缀和数组，求区间和需要循环一遍 $O(n)$。

如果有前缀和数组 $[l, r]$ 这一段数的和

就是 $S[r] - S[l-1]$。$O(1)$

↙

$S[r] = a_1 + a_2 + \cdots a_r$

$S[l-1] = a_1 + a_2 + \cdots a_{l-1}$

$S_0 = 0$ 的好处：处理边界

比如要求 $[1,1]$ 即 $S_{10} - S_0 = S_{10}$

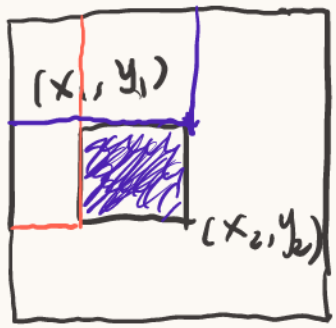
为了统一，所有都可以用 $S_r - S_{l-1}$ 这个公式。

二维前缀和：

a_{ij}

S_{ij}：以 (i,j) 这个点，左上角的有点的和

求矩阵中某个子矩阵的和：



求和：紫色区域。

=

$S_{x_2 y_2} - S_{x_1-1\ y_2} - S_{x_2\ y_1-1} + S_{x_1-1\ y_1-1}$

差分：

$a_1, a_2 \cdots a_n$

构造 $b_1, b_2 \cdots b_n$

使得 $a_i = b_1 + b_2 + \cdots + b_i$

b 就称为 a 的差分，a 就称为 b 的前缀和

$b_1 = a_1$

$b_2 = a_2 - a_1$

$b_3 = a_3 - a_2$

$\vdots$

$b_n = a_n - a_{n-1}$

作用：如果有 b 数组，我们就可以在 O(n) 的时间得到 a 数组（求一遍前缀和）

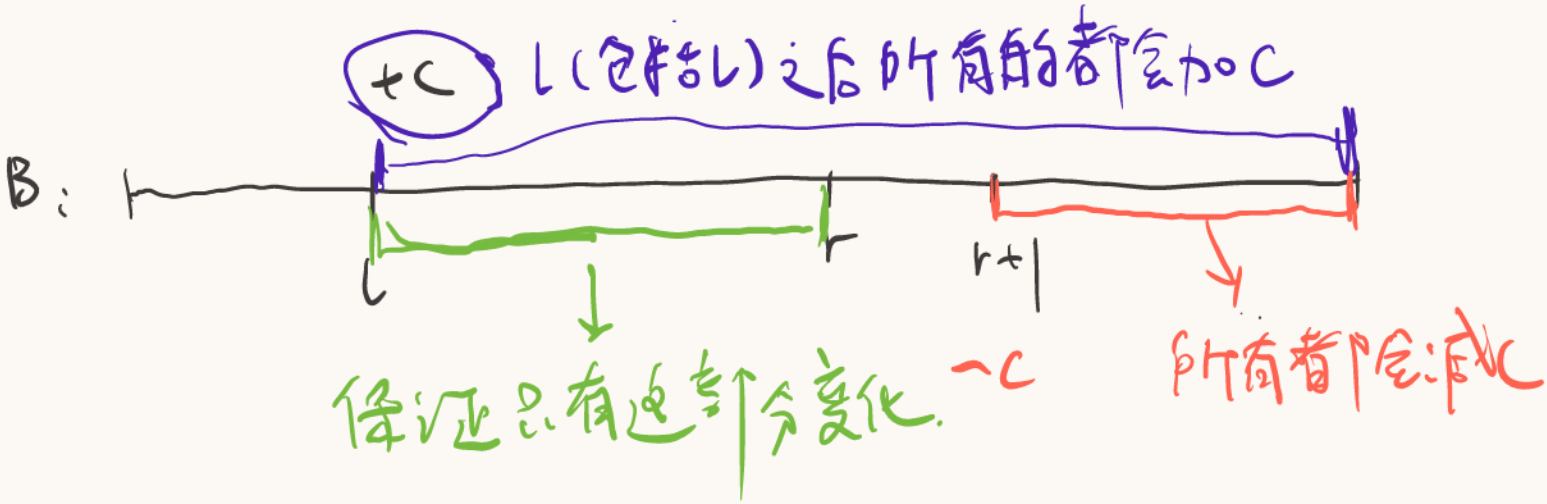
⇓

当要把原 a 数组中所有数都增加 c，暴力的做法是 for 一遍 O(n)。

要把 a 中 [l,r] 所有数都加上 c

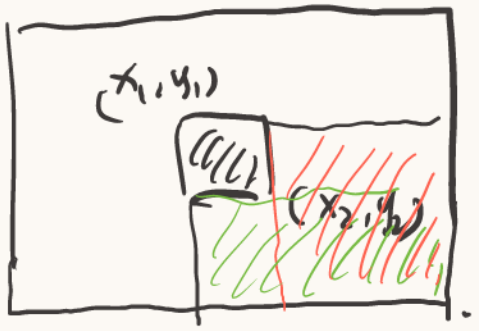
⇓

1. 要把 $b_l + c$，那么 $a_l, a_{l+1} \cdots a_n$ 的所有数都会加上 c，（因为 $a_{l+1} = a_l + b_l$），再让 $b_{r+1} - c$ 就可以了 O(1)



二维差分

a[i][j]



差分矩阵:

b[i][j] 给某个区域的值加b c.

$$\begin{cases} b[x_1][y_1] += c \\ b[x_2+1][y_1] -= c \\ b[x_1][y_2+1] -= c \\ b[x_2+1][y_2+1] += c \end{cases}$$

O(n) → O(1)