前缀和与差分

sdsy_ning

一、一维前缀和

- 对于序列 a[1...n]
- 前缀和：前 i 个数的和 sum[i] = a[1] + a[2] + ... + a[i] = sum[i-1] + a[i]
- 可求区间 [l, r] 的和：s(l,r) = a[l]+a[l+1]+...+a[r] = sum[r] - sum[l-1]
- 可求区间的 [l, r] 的异或和：（异或的逆运算是它本身）所以即为 sum[r] ^ sum[l-1]

补充

- 异或的逆运算是它本身

- $a * b = c$ ------> $a = c / b$ $(b!=0)$
- $a + b = c$ ------> $a = c - b$
- $a \oplus b = c$ ------> $a = c \oplus b$

差分

- a[1] a[2] ... a[n]
- b[1] b[2] ... b[n]

- b[1]=a[1]
- b[i]=a[i]-a[i-1]

- “前缀和”与“差分”是一对互逆运算
- a的差分序列是b
- b的前缀和序列是a
- 前缀和序列sum的差分序列也是a

- 让序列a的区间[l,r]里的数均+x
- 就是让原数列的差分数组b做两个点的修改：b[l]+=x;b[r+1]−=x
- 其他不变

- “区间操作” --------> “单点操作”

- 有了差分数组，原数组就不要了
- 如果想恢复原数组，只需要对差分数组求前缀和即可

#1375. [CSP2022入门级补赛] 第1题 植树节

#531. [USACO 2007 JAN] Tallest Cow

#534. [BZOJ 3043] [AcWing 100] IncDec Sequence 增减序列

二、二维前缀和

- 二维前缀和是根据容斥原理推导得到的：
- sum[i][j]=sum[i−1][j]+sum[i][j−1]−sum[i−1][j−1]+a[i][j]
- sum[i][j]表示的是 (1,1) 为左上角， (i,j) 为右下角的一个子矩形内的元素和

#656. [HNOI 2003] 激光炸弹

二维差分

对于一个给定的二维数组 arr，它的二维差分数组 d 中 **d[i][j] 可以用如下公式计算**：

① $d[0][0] = arr[0][0], i, j == 0$

② $d[0][j] = arr[0][j] - arr[0][j-1], i == 0, j >= 1$

③ $d[i][0] = arr[i][0] - arr[i-1][0], i >= 1, j == 0$

④ $d[i][j] = arr[i][j] - arr[i-1][j] - arr[i][j-1] + arr[i-1][j-1], i, j >= 1$

实际上，上面的公式是通过**二维数组 arr 是二维差分数组的前缀和**这个条件推导出来的，因此，不像一维差分定义那样直观。

二维前缀和的修改和求和

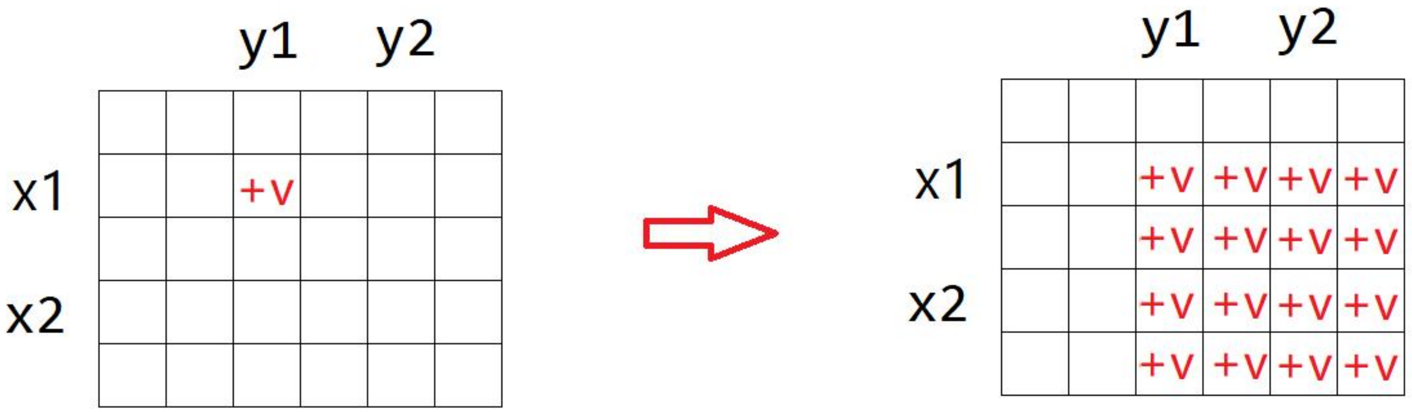
- 如果要使矩阵 x1<=x<=x2， y1<=y<=y2 里的数全部 +v 那么需要进行下面的操作（利用差分思想）：

- void add(ll x1,ll y1,ll x2,ll y2)
- {
- b[x1][y1]+=v; b[x2+1][y2+1]+=v; b[x1][y2+1]-=v; b[x2+1][y1]-=v;
- }

- 注意运用前缀和修改矩阵以后要先求一遍前缀和还原原矩阵，然后再求一遍前缀和才是真正的前缀和，这样才能继续进行下一步的操作

①

二维差分　　　　二维差分的前缀和



d[x1][y1] += v

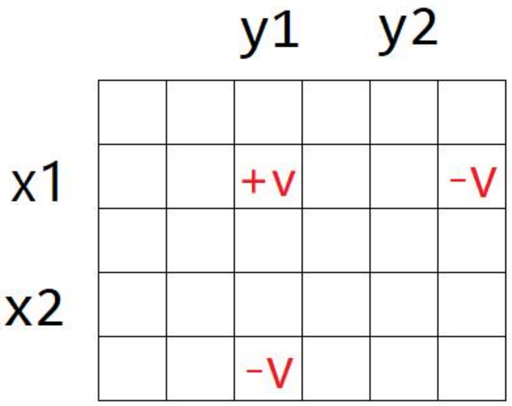
二维差分　　　　　　　　二维差分的前缀和

②

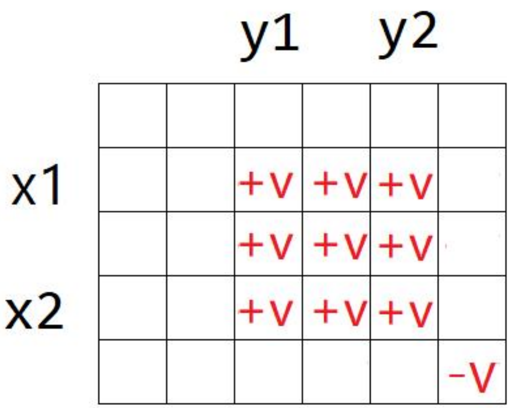
1. d[x1][y1] += v
2. d[x2 + 1][y1] -= v

③

二维差分



二维差分的前缀和



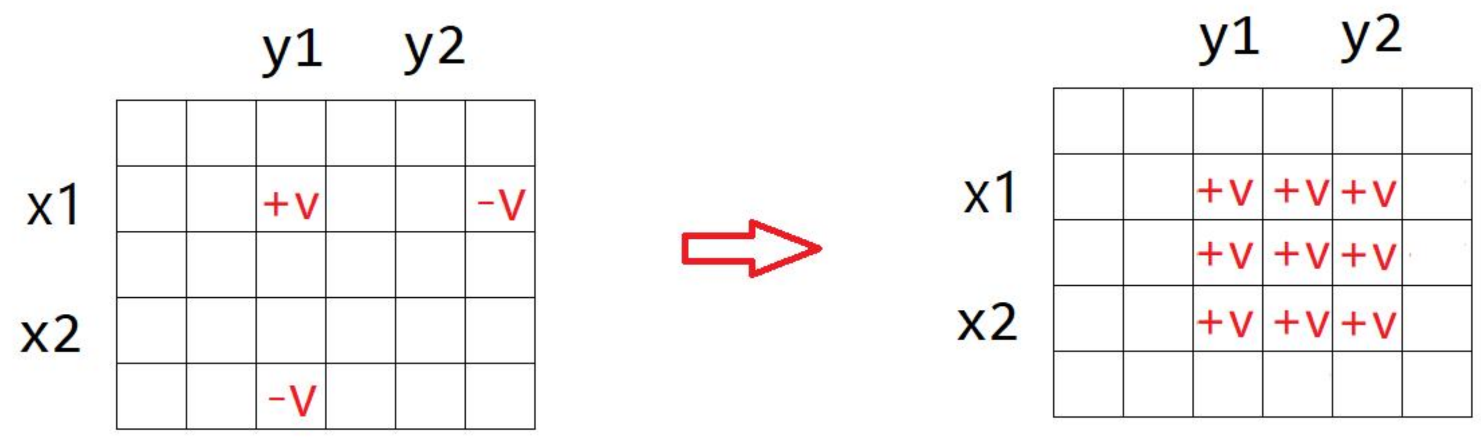
1. d[x1][y1] += v

2. d[x2 + 1][y1] -= v

3. d[x1][y2 + 1] -= v

二维差分　　　　二维差分的前缀和

④



1. d[x1][y1] += v
2. d[x2 + 1][y1] -= v
3. d[x1][y2 + 1] -= v
4. d[x2 + 1][y2 + 1] += v

#668. 牛妹吃豆子

#1379. [BZOJ2241] [SDOI2011]打地鼠