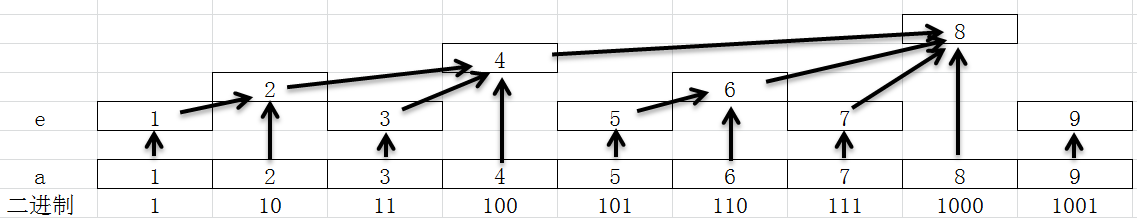
# 树状数组总结

**树状数组是一个类似于线段树的树状结构，它通过存储一定区间内的元素来达到快速插入、快速求和的效果。并且可以节省时间，节省空间，减少代码量，可谓是非常优秀的一个树形结构**

****

**方格中数字代表对应数组的第几个元素,下排是a数组,其上方的是e数组,最下的二进制则是对应编号的二进制表示.**

**可以观察到，树状数组把一个数组内的元素进行了一定的二分，并存储某些元素的和，使树的深度达到了logn，这样不论是对于查找还是插入都进行了极大的优化**

**每个数所指向的元素个数为1+它的二进制末尾0的个数我们把二进制中最后一个一出现的位置叫做lowbit，每个元素指向的下一个元素为x+lowbit(x)，现在要修改整棵树就比较容易了。**

1. **累加操作**

***void add(int x,int t)***

***{***

***while(x<=n){***

***e[x]+=t;***

***x=x+lowbit(x);***

***}***

***}***

1. **求和操作**

***int sum(int x)***

***{***

***int s=0;***

***while(x>0)***

***{***

***s+=e[x];***

***x-=lowbit(x);***

***}***

***return s;***

***}***

**其中，求和操作求的是1~x个元素之和，要求a~b个元素之和，只需：**

**Ans=sum(b)-sum(a-1);**

**其中运用了两次查询，对于查询单个元素，还可以优化为一次：**

**借助data[x]=e[x]-(query(x-1)-query(LCA(x,x-1)))**

**Int sum(x){**

**int ans=e[x];**

**int lca=x-lowbit(x);//求最近公共祖先**

**x--;**

**while(x!=lca){**

**ans-=e[x];**

**x-=e[x];**

**}**

**}**

**拓展操作：**

1. **区间修改、点查询**

**令e[i]为i元素与i-1元素的差值**

**那么区间[i,j]修改：**

**e[i]+=d; e[j+1]-=d;**

**单个点的求值：data[i]=e[1]+e[2]+…+e[i];**

1. **找第k小的元素**

**e[i]数组存取i出现的次数（有时数据太大可以离散优化）**

**问题可以转化为求区间和为k所对应的下标**

**区间和为递增区间，用二分查找**

**如4 6 5 5 2 2 3 1，每个数作为下标，add一次**

**然后sum(i)求出的就是当前1~i区间的数字个数，并且数字有序，二分即可**

1. **向高维拓展**

**树状数组与线段树相比，还是比较方便向高维拓展的，只需加一重循环即可**

**Void Change(int x,int y,int data){**

**While(x<=n){**

**Int ty=y;**

**While(ty<=n){**

**e[x][ty]+=delta;**

**ty+=lowbit(y);**

**}**

**x+=lowbit(x);**

**}**

**}**

**树状数组**

Ultra-QuickSort

**题目（poj 2299）**

请分析，对于一串数列，各个数字不相同，至少要交换多少次才能使该数列有序（从小到大）？

**输入：**

输入包含多组测试样例，每组测试样例都是先输入一个整数n<500000——序列的长度，以下n行每行包括一个整数 0 ≤ a[i] ≤ 999,999,999, 已输入的n为0作为输入的终止条件。

**输出：**

对于每一组样例，输出一个整数，表示至少要交换的次数。

**输入样例：**

5

9

1

0

5

4

3

1

2

3

0

**输出样例：**

6

0

**很标准的一道题，先将所有元素取出排序，去重，作为离散优化二分的依据**

**依次从第一位开始向树状数组对应位置加值，add(find(x),1)，并且同时算出当前比该元素大的值有多少，更新答案**

**Poj3928 Ping Pong**

题意：

n（3 < = 20000）乒乓球运动员住在一个西大街（考虑街道作为一个线段）。每个球员都有一个独特的技能等级。为了提高他们的技能等级，他们经常互相竞争。如果两个球员想参加比赛，他们必须选择其他乒乓球运动员的裁判，并在裁判的房子举行比赛。因为某些原因，参赛者不能选择一个技能等级较高或低于他们两个级别的裁判。参赛者必须走到裁判的房子，因为他们是懒惰的，他们想使他们的总步行距离不超过他们的房子之间的距离。当然，所有的球员都住在不同的房子，他们的房子的位置都是不同的。如果裁判或任何两名选手是不同的，我们称之为两个不同的游戏。现在的问题是：有多少不同的游戏可以在这个乒乓街举行？

输入

The first line of the input contains an integer T(1<=T<=20), indicating the number of test cases, followed by T lines each of which describes a test case.

输入的第一行包含一个整数t（1 < = T < = 20），表示测试用例的数量，然后是T行，其中每一个描述了一个测试用例。

Every test case consists of N + 1 integers. The first integer is N, the number of players. Then N distinct integers a1, a2 ... aN follow, indicating the skill rank of each player, in the order of west to east. (1 <= ai <= 100000, i = 1 ... N).

每一个测试用例都是由n个1个整数组成的。第一个整数是n，玩家的数量。然后，n个不同的整数A1、A2…一个跟随，指示每个玩家的技能等级，在西到东的顺序。（1 < = 100000，I = 1…n）。

Output

输出

For each test case, output a single line contains an integer, the total number of different games.

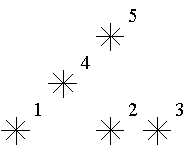
对于每一个测试用例，输出一个单行包含一个整数，不同游戏的总数。

**先把所有选手的能力值排序，然后加入能力值最低的选手，然后枚举所有可能成为裁判的选手，同时计算出这个裁判的左边和右边比裁判能力值大和小的值，ans+=(lmin\*rmax+lmax\*rmin);**

Stars

**题目（POJ 2352）**

天文学家常常检查星星地图，星星都有它的x,y坐标，星星的等级的是由下边星星数量和星星右边的星星数量决定。



例如,看看上面的星图。星星5的等级为3 (由星星1、2和4决定的)。星星2的等级为1（由星星1决定的）。在这张地图上0级的星星有一颗，1级的星星有两颗，2级的星星有一颗，3级的星星有一颗，

你要编写一个程序,计算每个等级的星星的数量。

**输入：**

第一行为星星的数量N（(1<=N<=15000)

接下来N行，每行为星星的x,y坐标，用空格来分开(0<=X,Y<=32000)，每一个点上只有一个星星，按Y的升序给出坐标，如果Y相同，则按照X的升序给出。

**输出：**

输出应该包含N行,每行一个数字。第一行0级星星的数量,第二行1级星星的数量等等,最后一行包含n – 1星星的数量。

**输入样例：**

5

1 1

5 1

7 1

3 3

5 5

**输出样例：**

1

2

1

1

0

**乍一看像一道二位树状数组题，其实是一维的，因为数据本来就是按y的大小排序，添加每个点x之前先计算sum(x)，这就是星星的等级，加入答案，继续直到循环结束**

**Hdu 4031 attack**

**问题描述**

今年是911的十周年，基地组织又想攻击美国了，美国有建设了一堵墙来保护自己，而基地组织有一个超级武器，每秒钟可以攻击连续的墙。而美国还有能量护盾来保护墙，每个能量护盾覆盖在一个单位长度的墙上，能抵挡攻击，但是每个能量护盾防御了一次攻击都需要T秒时间冷却下来，冷却期间，它不能抵挡攻击，比如该单位初盾牌抵挡了第k次攻击，那么它不能抵挡第k+1~(k+t-1)次攻击，过后，他们自动继续防御。

在战争中，知己知彼是非常重要的，因此指挥官想知道墙的某一部分被成功攻击了多少次，成功攻击就意味着盾牌没有防御到

**输入**

第一行T(<=20)，表示测试数据组数

每组测试数据第一行：N,Q,T，分别表示墙的长度，Q次攻击或询问，T秒的冷却时间

接下来Q行，格式有两种：

Attack si ti

攻击 si 到 ti 的墙. 1 ≤ si ≤ ti ≤ N

Query p

询问p位置的墙：1 ≤ p ≤ N

1 ≤ N, Q ≤ 20000  
1 ≤ t ≤ 50

**输出**

每组数据第一行：

Case i:

接下来是所有询问的答案，每个询问答案一行

**样例输入**

2

3 7 2

Attack 1 2

Query 2

Attack 2 3

Query 2

Attack 1 3

Query 1

Query 3

9 7 3

Attack 5 5

Attack 4 6

Attack 3 7

Attack 2 8

Attack 1 9

Query 5

Query 3

**样例输出**

Case 1:

0

1

0

1

Case 2:

3

2

**暂时没A，先放这里吧~**

**Poj 2985 The k-th Largest Group**

**题意**

有N只猫，M个操作。操作种类如下：

0 a b：把a b猫所在的组合并

1 k： 第K大的组的大小是多少。

**求第k大的数**

**Poj 1195 Mobile phones**

**题意：**

给你一个矩阵（初始化为0）和一些操作：

1 x y a表示在arr[x][y]加上a

2 l b r t 表示求左上角为(l,b),右下角为(r,t)的矩阵的和。

**高维树状数组即可**