# 寒假集训七题解

这次的题目都是偏数据结构的，前4题都是数据结构比较裸的应用，所以不是有很多可说的东西，大家需要自己搜这类数据结构的用法，建议采用刘汝佳书上的代码。

**Problem 1 poj 1611**

**The Suspects**

问题描述：

有很多组学生，在同一个组的学生经常会接触，也会有新的同学的加入。但是SARS是很容易传染的，只要在改组有一位同学感染SARS，那么该组的所有同学都被认为得了SARS。现在的任务是计算出有多少位学生感染SARS了。假定编号为0的同学是得了SARS的。

**Input**

The input file contains several cases. Each test case begins with two integers n and m in a line, where n is the number of students, and m is the number of groups. You may assume that 0 < n <= 30000 and 0 <= m <= 500. Every student is numbered by a unique integer between 0 and n−1, and initially student 0 is recognized as a suspect in all the cases. This line is followed by m member lists of the groups, one line per group. Each line begins with an integer k by itself representing the number of members in the group. Following the number of members, there are k integers representing the students in this group. All the integers in a line are separated by at least one space.   
A case with n = 0 and m = 0 indicates the end of the input, and need not be processed.

**Output**

For each case, output the number of suspects in one line.

**Sample Input**

100 4

2 1 2

5 10 13 11 12 14

2 0 1

2 99 2

200 2

1 5

5 1 2 3 4 5

1 0

0 0

**Sample Output**

4

1

1

并查集的题目，把一组的同学关联起来，然后查找和0是同一个祖先的元素的个数（包括0），统计结果就是答案。（多简洁。。。）

并查集就是一个个集合的概念，我一般用一个fa[]数组表示，其中fa[i]表示i元素的祖先，拥有共同祖先的元素说明他们属于同一个集合。这题就是把这些元素按关联分类，其中和0同类的元素就是SARS的可能患者。初始时每个人各自属于一个集合，fa[i]=i；如果i和j关联，说明他们有同一个祖先，可以让fa[i]=fa[j]或者fa[j]=fa[i]（这些都是大概的说，具体操作建议百度，有比较模板的代码）。

**Problem 2 poj 2524**

**Ubiquitous Religions**

问题描述：已知有n个大学生，其中有m对宗教信仰相同的学生，请你估算这n个学生中最多有多少种宗教信仰。

**Input**

The input consists of a number of cases. Each case starts with a line specifying the integers n and m. The next m lines each consists of two integers i and j, specifying that students i and j believe in the same religion. The students are numbered 1 to n. The end of input is specified by a line in which n = m = 0.

**Output**

For each test case, print on a single line the case number (starting with 1) followed by the maximum number of different religions that the students in the university believe in.

**Sample Input**

10 9

1 2

1 3

1 4

1 5

1 6

1 7

1 8

1 9

1 10

10 4

2 3

4 5

4 8

5 8

0 0

**Sample Output**

Case 1: 1

Case 2: 7

这里还是并查集的题目，也是比较简单的题目（感觉比第一题还简单），直接关联每2个点，然后遍历n个点。初始时fa[i]=i,每个元素独立一个集合，若干操作后，如果fa[i]！=i，表明他在别人的集合里，当fa[i]==i时，ans+1；输出ans即可。

**Problem 3 hdu 1754**

I Hate It

**Problem Description**

很多学校流行一种比较的习惯。老师们很喜欢询问，从某某到某某当中，分数最高的是多少。  
这让很多学生很反感。  
不管你喜不喜欢，现在需要你做的是，就是按照老师的要求，写一个程序，模拟老师的询问。当然，老师有时候需要更新某位同学的成绩。

**Input**

本题目包含多组测试，请处理到文件结束。  
在每个测试的第一行，有两个正整数 N 和 M ( 0<N<=200000,0<M<5000 )，分别代表学生的数目和操作的数目。  
学生ID编号分别从1编到N。  
第二行包含N个整数，代表这N个学生的初始成绩，其中第i个数代表ID为i的学生的成绩。  
接下来有M行。每一行有一个字符 C (只取'Q'或'U') ，和两个正整数A，B。  
当C为'Q'的时候，表示这是一条询问操作，它询问ID从A到B(包括A,B)的学生当中，成绩最高的是多少。  
当C为'U'的时候，表示这是一条更新操作，要求把ID为A的学生的成绩更改为B。

**Output**

对于每一次询问操作，在一行里面输出最高成绩。

**Sample Input**

5 6

1 2 3 4 5

Q 1 5

U 3 6

Q 3 4

Q 4 5

U 2 9

Q 1 5

**Sample Output**

5

6

5

9

这题就是线段树的题目了，线段树的题目比较难一些，刚开始看的时候不是很理解，大体就是每个结点代表一个区间，可以想象成一个树的样子，每个结点维护若干个属性，最大值，最小值，和等。建议是先看代码自己试着模拟一下，然后慢慢理解意思，多些几题才会理解其中的意思的，然后再写就不要看代码了，试着自己写，错了再回头对照着改，就知道自己哪里不是很理解了。

每个结点都有左子结点和右子结点，两个结点的属性结合起来可以得出父节点的属性，且可以进行点修改，区间修改，询问等操作，并同时能维护各关联点的属性的正确性（即操作同时能修改关联点）。这题也是比较裸的，点修改+区间询问，维护每个区间的最大值，详细操作函数请百度之，这里提供一个书上代码好了.

int ql,qr;//查询[ql,qr]中的最小值

int query(int o,int L,int R)

{

int M=L+(R-L)/2,ans=0; //成绩最小为0

if(ql<=L&&R>=qr)return maxv[o];

if(ql<=M)ans=Max(ans,query(2\*o,L,M));

If(M<qr)ans=Max(ans,query(2\*o+1,M+1,R));

return ans;

}

int p,v;//修改maxv[p]=v;

void update(int o,int L,int R)

{

int M=L+(R-L)/2;

If(L==R)maxv[o]=v;//叶节点

Else

{

If(p>=M)update(o\*2,L,M);

Else update(o\*2+1,M+1,R);

maxv[o]=Max(maxv[o\*2],max[o\*2+1]);//进入这个结点后，表示这个区间是关联结点，修改的结点是他子节点，所以这个结点属性可能发生改变。

}

}

**Problem 4 hdu 1166**

敌兵布阵

**Problem Description**

C国的死对头A国这段时间正在进行军事演习，所以C国间谍头子Derek和他手下Tidy又开始忙乎了。A国在海岸线沿直线布置了N个工兵营地,Derek和Tidy的任务就是要监视这些工兵营地的活动情况。由于采取了某种先进的监测手段，所以每个工兵营地的人数C国都掌握的一清二楚,每个工兵营地的人数都有可能发生变动，可能增加或减少若干人手,但这些都逃不过C国的监视。  
中央情报局要研究敌人究竟演习什么战术,所以Tidy要随时向Derek汇报某一段连续的工兵营地一共有多少人,例如Derek问:“Tidy,马上汇报第3个营地到第10个营地共有多少人!”Tidy就要马上开始计算这一段的总人数并汇报。但敌兵营地的人数经常变动，而Derek每次询问的段都不一样，所以Tidy不得不每次都一个一个营地的去数，很快就精疲力尽了，Derek对Tidy的计算速度越来越不满:"你个死肥仔，算得这么慢，我炒你鱿鱼!”Tidy想：“你自己来算算看，这可真是一项累人的工作!我恨不得你炒我鱿鱼呢!”无奈之下，Tidy只好打电话向计算机专家Windbreaker求救,Windbreaker说：“死肥仔，叫你平时做多点acm题和看多点算法书，现在尝到苦果了吧!”Tidy说："我知错了。。。"但Windbreaker已经挂掉电话了。Tidy很苦恼，这么算他真的会崩溃的，聪明的读者，你能写个程序帮他完成这项工作吗？不过如果你的程序效率不够高的话，Tidy还是会受到Derek的责骂的.

**Input**

第一行一个整数T，表示有T组数据。  
每组数据第一行一个正整数N（N<=50000）,表示敌人有N个工兵营地，接下来有N个正整数,第i个正整数ai代表第i个工兵营地里开始时有ai个人（1<=ai<=50）。  
接下来每行有一条命令，命令有4种形式：  
(1) Add i j,i和j为正整数,表示第i个营地增加j个人（j不超过30）  
(2)Sub i j ,i和j为正整数,表示第i个营地减少j个人（j不超过30）;  
(3)Query i j ,i和j为正整数,i<=j，表示询问第i到第j个营地的总人数;  
(4)End 表示结束，这条命令在每组数据最后出现;  
每组数据最多有40000条命令

**Output**

对第i组数据,首先输出“Case i:”和回车,  
对于每个Query询问，输出一个整数并回车,表示询问的段中的总人数,这个数保持在int以内。

**Sample Input**

1

10

1 2 3 4 5 6 7 8 9 10

Query 1 3

Add 3 6

Query 2 7

Sub 10 2

Add 6 3

Query 3 10

End

**Sample Output**

Case 1:

6

33

59

这题其实是比较裸的树状数组的题目，只有两种树状数组的操作（add和sum），因为sub可以转化为add。树状数组也是一种固定的数据结构，用法比较固定，只有2个操作函数，至于计算位置的函数，不妨记着就好，还是同样的建议，先试着记忆，自己写，边理解边记，不理解就先记着，数据结构的题目一般都比较套路模板，尤其是这次的题都是比较基础的题，没有太多能讲解的地方，具体函数可以百度之。

int lowbit(int x)

{

return x&-x;

}

int sum(int x)

{

int ret=0;

while(x>0)

{

ret+=c[x];

x-=lowbit(x);

}

return ret;

}

void add(int x,int d)

{

while(x<=n)

{

c[x]+=d;

X+=lowbit(x);

}

}

**Problem 5 xoj 1518**

为什么xoj那么“渣”，我还要找里面的题呢？因为百度不到！

这题当成线段树附加题吧，还是有难度的。

**循环计数游戏**

**Description**

假设你现在被困在一个房间里，面前摆放着N个范围是0~9的循环计数器，以及N个对应的计数按钮。每次按下按钮它所对应的计数器数值会加1（若当前计数数值为9，则按下后变为0）。想要逃脱这个房间你只能按照提示找出一串密码，游戏的规则如下，系统会要求你进行M次操作：

1.每次给出两个整数A和B (1 <= A <= B <= N)，计算序号从A到B之间的所有计数器数值的和并记下来。

2.依次按下序号A到B之间的计数器对应的计数按钮。

在M次操作完成之后，你所记录下来的数值则是最终的开门密码。显然，你已经没有时间手动进行操作了，你需要编写程序计算出最终的密码。

**Input**

第一行包含两个整数N和M（1 <= 250000, 1 <= M <= 100000）.

第二行包含最初所有计数器的状态，由N个10进制数字表示，中间没有分隔符，依次表示计数器1,2,...,N的初始状态。

接下来M行每行包含两个整数，为每次操作所需的A和B

**Output**

输出M行，表示每次操作所记录的求和结果。

**Sample Input**

4 3  
1357  
1 2  
2 3  
1 4

**Sample Output**

4  
9  
20

**这题比较难，也是线段树的题目，是去年校赛预赛的题目，当时当场过的人只有4个，我线段树也不强，也是赛后看题解才会做的。这题要用到线段树的点修改（赋初值），区间修改（按下某个区间所有的按钮，循环计数），区间询问。关键点是每个区间要维护什么信息？才能每次O（logn）的求出某个特定区间的sum值。虽然要求的是和，但是每个按钮是循环计数0~9，直接求和显然无法表示一个按钮超过9以后的数值。这里维护的每个点代表区间的信息，用的是长度为10的数组，sum[o][i]结点o代表的区间内的值为i的按钮有多少个。而和就表示成∑i\*sum[o][i]，i=0~9。每次区间修改（按下这个区间的按钮）设置一个add[o]值，表示这个区间内的结点被按下多少次，即这些按钮值偏移了多少位置（比如按下2次，那么sum[o][i]值在按下2次后变成了sum[o][(i+2)%10]）。这里用到了区间修改的操作，比原来的点修改更复杂更难理解，建议先做点比较简单的线段树懒标记的题目，再做这题。**

**因为个人感觉比较难，对刚接触线段树的人不易理解，而且数据结构的题目关键是要懂得这东西怎么用，这是基础，然后难一点的题再学会怎么变形使用（这要求你是真的理解这东西了），变形技巧什么可以交，但是数据结构本身的理解比较难教，这题直接上代码帮助理解。**

**#include<stdio.h>**

**#define MAXN 250000**

**char c[MAXN];**

**int y1,y2,v,p,\_sum;**

**int sum[3\*MAXN][10],add[3\*MAXN];**

**void maintain(int o,int l,int r)**

**{**

**int i,a[10];**

**if( r > l )for(i=0;i<10;i++)sum[o][i] = sum[2\*o][i] + sum[2\*o+1][i];**

**for(i=0;i<10;i++)a[i] = sum[o][i];**

**for(i=0;i<10;i++)sum[o][(i+add[o])%10]=a[i];**

**if(l==r)add[o]=0;**

**}**

**void update(int o,int l,int r)**

**{**

**int m=l+(r-l)/2;**

**if(y1<=l&&r<=y2)add[o]++;**

**else**

**{**

**if(y1<=m)update(2\*o,l,m);**

**if(y2>m)update(2\*o+1,m+1,r);**

**}**

**maintain(o,l,r);**

**}**

**void query(int o,int l,int r,int adds)**

**{**

**int m=l+(r-l)/2,i;**

**if(y1<=l&&r<=y2)for(i=0;i<10;i++)\_sum+=sum[o][i]\*((i+adds)%10);**

**else**

**{**

**if(y1<=m)query(2\*o,l,m,adds+add[o]);**

**if(y2>m)query(2\*o+1,m+1,r,adds+add[o]);**

**}**

**}**

**void create(int o,int l,int r)**

**{**

**int m=l+(r-l)/2,i;**

**if(l==r)sum[o][v]=1;**

**else**

**{**

**if(p<=m)create(2\*o,l,m);**

**else create(2\*o+1,m+1,r);**

**for(i=0;i<10;i++)sum[o][i]=sum[2\*o][i]+sum[2\*o+1][i];**

**}**

**}**

**int main()**

**{**

**int n,m,i;**

**scanf("%d %d %s",&n,&m,c);**

**for(i=0;i<n;i++)**

**{**

**v=c[i]-'0';**

**p=i+1;**

**create(1,1,n);**

**}**

**for(i=0;i<m;i++)**

**{**

**scanf("%d%d",&y1,&y2);**

**\_sum=0;**

**query(1,1,n,0);**

**printf("%d\n",\_sum);**

**update(1,1,n);**

**}**

**return 0;**

**}**