By

r\_clover-任琪宇

2016.10.11

2016代码库

[杂七杂八 4](#_Toc470947000)

[命令行指令 4](#_Toc470947001)

[Bash 4](#_Toc470947002)

[long double 4](#_Toc470947003)

[bigLL 5](#_Toc470947004)

[Bitset 6](#_Toc470947005)

[java 7](#_Toc470947006)

[Java开根号 10](#_Toc470947007)

[模拟退火 13](#_Toc470947008)

[字符串 14](#_Toc470947009)

[AC自动机求模式串出现次数和 14](#_Toc470947010)

[shfit\_and以及几种读入读出 17](#_Toc470947011)

[kmp 18](#_Toc470947012)

[Manacher 19](#_Toc470947013)

[Trie and AC自动机 20](#_Toc470947014)

[字符串hash 22](#_Toc470947015)

[最小表示法 23](#_Toc470947016)

[后缀数组 24](#_Toc470947017)

[后缀自动机 27](#_Toc470947018)

[数据结构 30](#_Toc470947019)

[2D\_bit 30](#_Toc470947020)

[CDQ分治 33](#_Toc470947021)

[Splay 46](#_Toc470947022)

[AAA树 51](#_Toc470947023)

[st表求区间最大值与和 56](#_Toc470947024)

[叉姐线段树 57](#_Toc470947025)

[插头dp 58](#_Toc470947026)

[插头dp(2) 62](#_Toc470947027)

[莫队算法 65](#_Toc470947028)

[带修改莫队 69](#_Toc470947029)

[整体二分 71](#_Toc470947030)

[静态主席树 78](#_Toc470947031)

[动态主席树 80](#_Toc470947032)

[可持续化并查集 85](#_Toc470947033)

[多重背包 88](#_Toc470947034)

[二维st表 89](#_Toc470947035)

[康拓展开 91](#_Toc470947036)

[数位dp（考虑前导零） 93](#_Toc470947037)

[四边形不等式 94](#_Toc470947038)

[斜率优化dp 95](#_Toc470947039)

[舞蹈链 97](#_Toc470947040)

[左偏树 101](#_Toc470947041)

[kdtree带修改 103](#_Toc470947042)

[树 106](#_Toc470947043)

[树分治 106](#_Toc470947044)

[树链剖分 108](#_Toc470947045)

[树上莫队（附树上启发式合并） 117](#_Toc470947046)

[基环外向树 127](#_Toc470947047)

[虚树 131](#_Toc470947048)

[数论 135](#_Toc470947049)

[快速幂与和 135](#_Toc470947050)

[Fft 136](#_Toc470947051)

[Fwt 139](#_Toc470947052)

[Ntt 140](#_Toc470947053)

[Lucas 144](#_Toc470947054)

[lucas推广（数论小板子） 145](#_Toc470947055)

[n以内素数 147](#_Toc470947056)

[Schreier-Sims 151](#_Toc470947057)

[斯特林数 154](#_Toc470947058)

[五边形数 155](#_Toc470947059)

[线性基 158](#_Toc470947060)

[圆的反演 160](#_Toc470947061)

[单纯型法 162](#_Toc470947062)

[欧拉函数前缀和 165](#_Toc470947063)

[矩阵的逆 166](#_Toc470947064)

[集合操作 169](#_Toc470947065)

# 杂七杂八

## 命令行指令

命令行指令

g++ -o r r.cpp

./r

chmod +x test.sh

./test.sh

javac Main.java

java Main

## Bash

echo begin

#!/bin/becho beginash

while true; do

./r > input

./a < input > outputa

./b < input > outputb

date

#echo finish one cmpare

diff outputa outputb

if [ $? -ne 0 ] ; then break; fi

done

## long double

#include < iomanip.h>

int main()

{

// freopen("in.txt","r",stdin);

while(cin>>C>>M>>F){

if( M > 1e6 || M > C )

cout<<1.0<<endl;

else if(M==1) cout<<0<<endl;

else{

long double ans = 1;

for(int i=1;i<=M-1;i++)

ans \*= 1.0 - i/((long double)2\*C-i);

cout<<std::fixed<<setprecision(20)<<1-ans<<endl;

}

}

return 0;

}

## bigLL

#define LL long long

const LL BIGINF = 1e18;

struct bigLL

{

LL p , big;

bigLL(){};

bigLL(LL x,LL b=0):p{x},big(b){};

bigLL operator + ( const bigLL & b ) const{

bigLL c( p+b.p , big+b.big );

c.big += c.p / BIGINF;

c.p %= BIGINF;

}

void print()

{

if( big )

{

printf("%lld",big);

LL t = BIGINF;

while( t>1 )

{

printf("%lld",(p%t)/(t/10));

t/=10;

}

}

else printf("%lld",p);

}

}

## Bitset

#include <bitset>//正常情况 加速32倍 可能还要加速更多

不仅要注意时间上的优化，还要注意空间时候否可行。不可行时可用分块来做。

计算时间 printf("size is %dMB\n",(sizeof a)>>20);

bitset<n>p;

b.any() b中是否存在置为1的二进制位？

b.none() b中不存在置为1的二进制位吗？

b.count() b中置为1的二进制位的个数

b.size() b中二进制位的个数

b[pos] 访问b中在pos处的二进制位

b.test(pos) b中在pos处的二进制位是否为1？

b.set() 把b中所有二进制位都置为1

b.set(pos) 把b中在pos处的二进制位置为1

b.reset() 把b中所有二进制位都置为0

b.reset(pos) 把b中在pos处的二进制位置为0

b.flip() 把b中所有二进制位逐位取反

b.flip(pos) 把b中在pos处的二进制位取反

b.to\_ulong() 用b中同样的二进制位返回一个unsigned long值

b.\_Find\_first();

b.\_Find\_next(pos);

os << b 把b中的位集输出到os流

以及所有位操作。

＃include <bitset>

using std::bitset;

bitset<32> bitvec; //32位，全为0。

bitset<n> b; b有n位，每位都为0

bitset<n> b(u); b是unsigned long型u的一个副本

bitset<n> b(s); b是string对象s中含有的位串的副本

bitset<n> b(s, pos, n); b是s中从位置pos开始的n个位的副本

hdu 5413 n 2e4

普通 n^2 111.363s

bitset 2.385s

## java

框架：

import java.io.\*;

import java.util.\*;

import java.math.\*;

import java.text.\*;

public class Main

public static void main(String args[]) throws Exception

{

Scanner cin = new Scanner(new BufferedInputStream(System.in));

//Scanner cin=new Scanner(new File("text.txt"));

PrintWriter cout = new PrintWriter(new BufferedOutputStream(System.out));

//PrintWriter cout = new PrintWriter(new File("output.txt"));

/\*\*

主体在这里

\*\*/

/\*\*

int T=cin.nextInt();

while(T-->0)

{

}

\*\*/

/\*\*

while(cin.hasNext())

{

}

\*\*/

cout.flush();

}

}

注意：

交题时千万不要交成c++！！！ T\_T;

1.建议java全部使用显示类型转换，不支持其他类型到boolean的转换。

2. PrintWriter cout = new PrintWriter(new BufferedOutputStreamSystem.out));

使用了输出缓冲区，当没有cout.flush();时不会输出（道理同cout的 cout<<endl;），

调试时要注意。

3.java 字符串数组非常难用，建议一律使用String。

4.对于输出浮点数保留几位小数的问题，可以使用DecimalFormat类，

import java.text.\*;

DecimalFormat f = new DecimalFormat("#.00#");

DecimalFormat g = new DecimalFormat("0.000");

double a = 123.45678, b = 0.12;

System.out.println(f.format(a));

System.out.println(f.format(b));

System.out.println(g.format(b));

这里0指一位数字，#指除0以外的数字。

（这种方式对BigDecimal无效）

BigDecimal ：四舍五入

d = d . setScale( scale , BigDecimal.ROUND\_HALF\_UP ) ;

5.下面在java.util包里Arrays类的几个方法可替代C/C++里的memset、 qsort/sort 和 bsearch:

Arrays.fill()

Arrays.sort()

Arrays.binarySearch()

6.所有函数，全局变量一律加上static，省事。

使用BigInteger，BigDecimal：

1.建议写上

static BigInteger bi(int n)

{

return BigInteger.valueOf((long)n);

}

static BigInteger bi(String s)

{

return new BigInteger(s);

}

static BigDecimal bd(String s)

{

return new BigDecimal(s);

}

static BigDecimal bd(double n)

{

return BigDecimal.valueOf(n);

}

不然你会被累死的。

2.比较时

用这个 p.equals(bi(0L));

不能用p.intValue()==0;

3.BigDecimal输出时

c.stripTrailingZeros().toPlainString()

stripTrailingZeros()去末位零。

toPlainString() 防止输出诸如1e-9

4.诸如add（）一类函数，不对原值进行改变，是返回值进行改变。

要写成p=p.add(q);

## Java开根号

使用牛顿迭代法

import java.io.\*;

import java.util.\*;

import java.math.\*;

import java.text.\*;

public class Main {

static Scanner cin = new Scanner(new BufferedInputStream(System.in));

static PrintWriter cout = new PrintWriter(new BufferedOutputStream(System.out));

public static void main(String args[]) throws Exception{

BigInteger n = new BigInteger("5").multiply(BigInteger.TEN.pow(400));

BigInteger \_5 = BigInteger.TEN.pow(200) , tmp = n;

BigInteger \_2 = new BigInteger("2");

while( tmp.compareTo(\_5) != 0 ){

tmp = \_5;

\_5 = \_5.add(n.divide(\_5)).divide(\_2);

}

BigDecimal hua = new BigDecimal(\_5.toString());

hua = hua.divide(BigDecimal.TEN.pow(200));

hua = hua.subtract(BigDecimal.ONE).divide(BigDecimal.valueOf(2));

while( cin.hasNext() ){

BigDecimal a = new BigDecimal(cin.next());

BigDecimal b = new BigDecimal(cin.next());

int t = 0;

if( a.compareTo(BigDecimal.ZERO) == 0 ) t++;

if( b.compareTo(BigDecimal.ZERO) == 0 ) t++;

if( t == 2 )

cout.println(0);

else if( t == 1 )

cout.println(1);

else{

if( b.compareTo(hua.multiply(a).setScale(0,BigDecimal.ROUND\_HALF\_UP)) == 0 )

cout.println(0);

else if( a.compareTo(hua.multiply(b).setScale(0,BigDecimal.ROUND\_HALF\_UP)) == 0 )

cout.println(0);

else

cout.println(1);

}

}

cout.flush();

}

}

java.math.BigDecimal

不可变的、任意精度的有符号十进制数。BigDecimal 由任意精度的整数非标度值和32位的整数标度(scale)组成。

如果为零或正数，则标度是小数点后的位数。如果为负数，则将该数的非标度值乘以10的负scale次幂。

因此，BigDecimal表示的数值是(unscaledValue × 10-scale)。

与之相关的还有两个类：

java.math.MathContext：

该对象是封装上下文设置的不可变对象，它描述数字运算符的某些规则，如数据的精度，舍入方式等。

java.math.RoundingMode：

这是一种枚举类型，定义了很多常用的数据舍入方式。

这个类用起来还是很比较复杂的，原因在于舍入模式，数据运算规则太多太多，

不是数学专业出身的人看着中文API都难以理解，这些规则在实际中使用的时候在翻阅都来得及。

在银行、帐户、计费等领域，BigDecimal提供了精确的数值计算。其中8种舍入方式值得掌握。

1、ROUND\_UP

舍入远离零的舍入模式。

在丢弃非零部分之前始终增加数字(始终对非零舍弃部分前面的数字加1)。

注意，此舍入模式始终不会减少计算值的大小。

2、ROUND\_DOWN

接近零的舍入模式。

在丢弃某部分之前始终不增加数字(从不对舍弃部分前面的数字加1，即截短)。

注意，此舍入模式始终不会增加计算值的大小。

3、ROUND\_CEILING

接近正无穷大的舍入模式。

如果 BigDecimal 为正，则舍入行为与 ROUND\_UP 相同;

如果为负，则舍入行为与 ROUND\_DOWN 相同。

注意，此舍入模式始终不会减少计算值。

4、ROUND\_FLOOR

接近负无穷大的舍入模式。

如果 BigDecimal 为正，则舍入行为与 ROUND\_DOWN 相同;

如果为负，则舍入行为与 ROUND\_UP 相同。

注意，此舍入模式始终不会增加计算值。

5、ROUND\_HALF\_UP

向“最接近的”数字舍入，如果与两个相邻数字的距离相等，则为向上舍入的舍入模式。

如果舍弃部分 >= 0.5，则舍入行为与 ROUND\_UP 相同;否则舍入行为与 ROUND\_DOWN 相同。

注意，这是我们大多数人在小学时就学过的舍入模式(四舍五入)。

6、ROUND\_HALF\_DOWN

向“最接近的”数字舍入，如果与两个相邻数字的距离相等，则为上舍入的舍入模式。

如果舍弃部分 > 0.5，则舍入行为与 ROUND\_UP 相同;否则舍入行为与 ROUND\_DOWN 相同(五舍六入)。

7、ROUND\_HALF\_EVEN

向“最接近的”数字舍入，如果与两个相邻数字的距离相等，则向相邻的偶数舍入。

如果舍弃部分左边的数字为奇数，则舍入行为与 ROUND\_HALF\_UP 相同;

如果为偶数，则舍入行为与 ROUND\_HALF\_DOWN 相同。

注意，在重复进行一系列计算时，此舍入模式可以将累加错误减到最小。

此舍入模式也称为“银行家舍入法”，主要在美国使用。四舍六入，五分两种情况。

如果前一位为奇数，则入位，否则舍去。

以下例子为保留小数点1位，那么这种舍入方式下的结果。

1.15>1.2 1.25>1.2

8、ROUND\_UNNECESSARY

断言请求的操作具有精确的结果，因此不需要舍入。

如果对获得精确结果的操作指定此舍入模式，则抛出ArithmeticException。

不同舍入模式下的舍入操作汇总

根据给定的舍入模式将输入数字舍入为一位数的结果

输入数字 UP DOWN CEILING FLOOR HALF\_UP HALF\_DOWN HALF\_EVEN UNNECESSARY

5.5 6 5 6 5 6 5 6 抛出 ArithmeticException

2.5 3 2 3 2 3 2 2 抛出 ArithmeticException

1.6 2 1 2 1 2 2 2 抛出 ArithmeticException

1.1 2 1 2 1 1 1 1 抛出 ArithmeticException

1.0 1 1 1 1 1 1 1 1

-1.0 -1 -1 -1 -1 -1 -1 -1 -1

-1.1 -2 -1 -1 -2 -1 -1 -1 抛出 ArithmeticException

-1.6 -2 -1 -1 -2 -2 -2 -2 抛出 ArithmeticException

-2.5 -3 -2 -2 -3 -3 -2 -2 抛出 ArithmeticException

-5.5 -6 -5 -5 -6 -6 -5 -6 抛出 ArithmeticException

## 模拟退火

/\*

\* J(y)：在状态y时的评价函数值

\* Y(i)：表示当前状态

\* Y(i+1)：表示新的状态

\* r： 用于控制降温的快慢

\* T： 系统的温度，系统初始应该要处于一个高温的状态

\* T\_min ：温度的下限，若温度T达到T\_min，则停止搜索

\*/

while( T > T\_min )

{

dE = J( Y(i+1) ) - J( Y(i) ) ;

　　if ( dE >=0 ) //表达移动后得到更优解，则总是接受移动

Y(i+1) = Y(i) ; //接受从Y(i)到Y(i+1)的移动

　　else

　　{

// 函数exp( dE/T )的取值范围是(0,1) ，dE/T越大，则exp( dE/T )也

if ( exp( dE/T ) > random( 0 , 1 ) )

Y(i+1) = Y(i) ; //接受从Y(i)到Y(i+1)的移动

　　}

　　T = r \* T ; //降温退火 ，0<r<1 。r越大，降温越慢；r越小，降温越快

　　/\*

　　\* 若r过大，则搜索到全局最优解的可能会较高，但搜索的过程也就较长。若r过小，则搜索的过程会很快，但最终可能会达到一个局部最优值

　　\*/

　　i ++ ;

}

# 字符串

## AC自动机求模式串出现次数和

///普通AC自动机在求文本串T中模式串a[i]出现次数时\

时间复杂度为 O(len(T)+sumlen(a)+k) k为答案\

下面的自动机复杂度为 O(len(T)+sumlen(a)) 但只能求次数

#define cls(p) clr(p,0)

const int maxn = 1e5;

const int maxsize = 30;

struct Trie

{

int ch[maxn][maxnsize];

int val[maxn];

vector<int>vv[maxn];

int sz;

void init()

{

sz = 1;

cls(ch[0]);

vv[0].clear();

}

int idx(int c)

{

return c - 'a';

}

void insert(char \*s,int v)

{

int u = 0 , n = strlen(s);

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

{

int c = idx(s[i]);

if(!ch[u][c])

{

cls(ch[sz]);

val[sz]=0;

vv[sz].clear();

ch[u][c] = sz++;

}

u=ch[u][c];

}

val[u]=v;

vv.push\_back(v);

}

///AC

LL ans;

LL sum[maxn];

int last[maxn];

void print(int i,int j)

{

if(j)

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ans+=sum[j];

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

}

void find(char \*T)

{

ans.clear();

int n = strlen(T);

int j = 0;

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

{

int c = idx(T[i]);

while(j&&!ch[j][c]) j = f[j];

j=ch[j][c];

if(val[j]) print(i,j);

else if (last[j]) print(i,last[j]);

}

}

int getFail()

{

queue<int>q;

f[0]=0;

for(int c=0;c<maxsize;c++)

{

int u = ch[0][c];

if(u)

{

f[u]=0;

q.push(u);

last[u]=0;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

sum[u] = (val[u]?1:0) + sum[last[u]];

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

}

while(!q.empty())

{

int r = q.front(); q.pop();

for(int c=0;c<maxszie;c++)

{

int u=ch[r][c];

if(!u) continue;

q.push(u);

int v = f[r];

while(v&&!ch[v][c]) v = f[v];

f[u]=ch[v][c];

last[u] = val[f[u]]?f[u]:last[f[u]];

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

sum[u] = (val[u]?1:0) + sum[last[u]];

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

}

}

}

## shfit\_and以及几种读入读出

#include<cstdio>

#include<cmath>

#include<cstring>

#include<cstdlib>

#include<iostream>

#include<algorithm>

#include<vector>

#include<queue>

#include<map>

#include<complex>

#include<set>

#include<assert.h>

#include<bitset>

#define lson l,m,rt<<1

#define rson m+1,r,rt<<1|1

using namespace std;

#define ll long long

const int maxn = 6e6;

char s[maxn];

const int M = 1e3+20;

bitset<M>pa[20],pb;

int main()

{

freopen("in.txt","r",stdin);

//cout<<(ll)MAXN\*MAXN<<endl;

int n,t,x;

while(~scanf("%d",&n)){

for(int i=0;i<10;i++)

pa[i].reset();

pb.reset();

for(int i=0;i<n;i++){

scanf("%d",&t);

while(t--){

scanf("%d",&x);

pa[x].set(i);

}

}

scanf(" ");

/\*\*int len = fread(s,sizeof (char),maxn,stdin);

len--;

只有最后一个字符串时才可以用\*/

/\*\*读取一行

while(fgets(s,maxn,stdin)){

}\*/

fgets(s,maxn,stdin);

int len = strlen(s);

len--;

///两种读法都会把'\n'读进去

// cout<<len<<endl;

for(int i=0;i<len;i++){

pb = (pb<<1).set(0) & pa[s[i]-'0'];

if( pb.test(n-1) ){

char tmp = s[i+1];

s[i+1] = '\n';

// puts(s+i-n+1);

fwrite(s+i-n+1,sizeof (char),n+1,stdout);

///还有一个'\n'的位置，不需要写末尾0的位置。

s[i+1] = tmp;

}

}

}

return 0;

}

## kmp

vector<int>ans;

void getFail(char \*P,int \*f)

{

int m = strlen(P);

f[0]=0;

f[1]=0;

for(int i=1;i<m;i++)

{

int j = f[i];

while(j&&P[i]!=P[j]) j=f[j];

f[i+1]=P[i]==P[j]?j+1:0；

}

}

void find(char \*T,char \*P,int \*f)

{

int n = strlen(T),m= strlen(P);

getFail(P,f);

int j =0;

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

{

while(j&&P[j]!=T[i]) j=f[j];

if(P[j]==T[i]) j++;

if(j==m) ans.push\_back(i-m+1);

}

}

## Manacher

const int maxn = 1000\*3+100;

//至少两倍空间

char ma[maxn];

int mp[maxn],l;

///回文串长度mp[i]-1,开始位置i/2-mp[i]/2

void manacher(char \*s,int len){

l=0;

ma[l++]='$';

ma[l++]='#';

for(int i=0;i<len;i++)

{

ma[l++]=s[i];

ma[l++]='#';

}

ma[l]=0;

int mx=0,id=0;

for(int i=0;i<l;i++)

{

mp[i]=mx>i?min(mp[2\*id-i],mx-i):1;

while(ma[i+mp[i]]==ma[i-mp[i]]) mp[i]++;

if(i+mp[i]>mx)

{

mx=i+mp[i];

id=i;

}

}

}

## Trie and AC自动机

#define cls(p) clr(p,0)

const int maxn = 1e5;

const int maxsize = 30;

struct Trie

{

int ch[maxn][maxnsize];

int val[maxn];

vector<int>vv[maxn];

int sz;

void init()

{

sz = 1;

cls(ch[0]);

vv[0].clear();

}

int idx(int c)

{

return c - 'a';

}

void insert(char \*s,int v)

{

int u = 0 , n = strlen(s);

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

{

int c = idx(s[i]);

if(!ch[u][c])

{

cls(ch[sz]);

val[sz]=0;

vv[sz].clear();

ch[u][c] = sz++;

}

u=ch[u][c];

}

val[u]=v;

vv.push\_back(v);

}

///AC

vector<pair<int,int> >ans;

int last[maxn],f[maxn];

void print(int i,int j)

{

if(j)

{

ans.push\_back(make\_pair(i,j));

print(i,last[j]);

}

}

void find(char \*T)

{

ans.clear();

int n = strlen(T);

int j = 0;

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

{

int c = idx(T[i]);

while(j&&!ch[j][c]) j = f[j];

j=ch[j][c];

if(val[j]) print(i,j);

else if (last[j]) print(i,last[j]);

}

}

int getFail()

{

queue<int>q;

f[0]=0;

for(int c=0;c<maxsize;c++)

{

/\*\*\*\*\*\*\*\*\*\*/

int u = ch[0][c];

/\*\*\*\*\*\*\*\*\*\*/

if(u)

{

f[u]=0;

q.push(u);

last[u]=0;

}

}

while(!q.empty())

{

int r = q.front(); q.pop();

for(int c=0;c<maxszie;c++)

{

int u=ch[r][c];

if(!u) continue;

q.push(u);

int v = f[r];

while(v&&!ch[v][c]) v = f[v];

f[u]=ch[v][c];

last[u] = val[f[u]]?f[u]:last[f[u]];

}

}

}

}

## 字符串hash

#define ull unsigned long long

const ull x = 233;

ull pow\_x[maxn];

ull H[maxn];

char s[maxn];int len;

void Hash\_init(){

len = strlen(s);

pow\_x[0] = 1ull;

for(int i=1;i<=len;i++)

pow\_x[i] = pow\_x[i-1] \* x;

H[len]=0;

for(int i=len-1;i>=0;i--)

H[i] = H[i+1]\*x + (ull)s[i];

}

ull Hash(int i,int len){

return H[i] - H[i+len]\*pow\_x[len];

}

///偶尔会遇到卡ull的题，用这个大模数+快速承

#define ull unsigned long long

const ull mod = 1125899839733759ll;

const ull x = 9987;

ull pow\_x[maxn];

ull H[maxn];

ull mul(ull x,ull n){

ull r = 0;

while(n){

if(n&1) r = r+x;

if( r >= mod ) r -= mod;

x+=x;

if( x >= mod ) x -= mod;

n>>=1;

}

return r;

}

char s[maxn];int len;

void Hash\_init(){

len = strlen(s);

pow\_x[0] = 1ull;

for(int i=1;i<=len;i++)

pow\_x[i] = pow\_x[i-1] \* x % mod;

H[len]=0;

for(int i=len-1;i>=0;i--)

H[i] = (H[i+1]\*x + s[i])%mod;

}

ull Hash(int i,int len){

return (H[i] - mul(H[i+len],pow\_x[len]) + mod)%mod;

}

## 最小表示法

///只能寻找距离开头最近的字典序最小

int find(char \*s)

{

int n = strlen(s),p=0,q=1;

while(p<n&&q<n)

{

int i;

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

if( s[(p+i)%n] != s[(q+i)%n])

break;

if( s[(p+i)%n] > s[(q+i)%n] ) /// 字典序最小

///if( s[(p+i)%n] < s[(q+i)%n] ) /// 字典序最大

p = p + i + 1;

else

q = q + i + 1;

if(p==q) q++;

}

return min(p,q); ///返回的下标从0开始

}

## 后缀数组

#include<string.h>

const int maxn =1e5;

#define FOR(i,a,b) \

for(i=a; (a<b)?(i<=b):(i>=b) ; (a<b)?(i++):(i--) )

struct suffix\_array

{

char s[maxn];

int sa[maxn],t[maxn],t2[maxn],c[maxn];

int m,n;

///构造sa数组

void build\_sa()

{

int i,\*x=t,\*y=t2;

FOR(i,0,m-1) c[i]=0;

FOR(i,0,n-1) c[x[i]=s[i]]++;

FOR(i,1,m-1) c[i] += c[i-1];

FOR(i,n-1,0) sa[--c[x[i]]]=i;

for(int k=1;k<=n;k<<=1)

{

int p=0;

FOR(i,n-k,n-1) y[p++]=i;

FOR(i,0,n-1) if(sa[i]>=k) y[p++]=sa[i]-k;

FOR(i,0,m-1) c[i] = 0;

FOR(i,0,n-1) c[x[y[i]]]++;

FOR(i,0,m-1) c[i] += c[i-1];

FOR(i,n-1,0) sa[--c[x[y[i]]]] = y[i];

swap(x,y);

p=1;

x[sa[0]]=0;

FOR(i,1,n-1)

x[sa[i]] = y[sa[i-1]]==y[sa[i]] &&

y[sa[i-1]+k] == y[sa[i]+k] ? p-1 : p++;

if(p>=n) break;

m=p;

}

}

///匹配模式串

int len;

int cmp(char \*pa,int p)

{

return strncmp(pa,s+sa[p],len);

}

int find\_first(char \*P)

{

len = strlen(P);

if(cmp(P,0)<0) return -1;

if(cmp(P,n-1)>0) return -1;

int L=0,R=n-1,ans=n;

while(R>=L)

{

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

int res = cmp(P,M);

if(res<=0)

{

R=M-1;

if( res == 0 ) ans = min(ans,M);

}

else L=M+1;

}

if(ans==n) return -1;

else return ans;

}

int find\_last(char \*P)

{

len = strlen(P);

if(cmp(P,0)<0) return -1;

if(cmp(P,n-1)>0) return -1;

int L=0,R=n-1,ans=-1;

while(R>=L)

{

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

int res = cmp(P,M);

if(res>=0)

{

L=M+1;

if(res==0) ans = max(ans,M);

}else R = M-1;

}

return ans;

}

///构造rank，height数组

int rank[maxn],height[maxn];

void getHeight()

{

int i,j,k=0;

for(int i=0;i<n;i++) rank[sa[i]]=i;

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

{

if(k) k--;

int j=sa[rank[i]-1];

while(s[i+k]==s[j+k]) k++;

height[rank[i]]=k;

}

height[n] = 0;

}

int d[maxn][30],flog[maxn];

void RMQ\_init()

{

for(int i=0;i<n;i++) d[i][0]=height[i];

flog[0]=-1;

for(int i=1;i<n;i++) flog[i] = flog[i>>1]+1;

for(int j=1;(1<<j)<=n;j++)

for(int i=0;i+(1<<j)<=n;i++)

d[i][j] = min(d[i][j-1],d[i+(1<<(j-1))][j-1]);

}

int RMQ(int L,int R)

{

int k = flog[R-L+1];

return min(d[L][k],d[R-(1<<k)+1][k]);

}

int lcp(int j,int k)

{

if(j==k) return n-k;

if(rank[j]>rank[k]) swap(j,k);

return RMQ(rank[j]+1,rank[k]);

}

void init(char \*ss,int mm=200)

{

strcpy(s,ss);

n=strlen(s)+1;

m=mm;

}

int all()

{

build\_sa();

getHeight();

RMQ\_init();

}

}sp;

## 后缀自动机

const int maxn = 9e5 \* 5;

///后缀自动机最少要开两倍空间

///可以处理一切增量形式的字符串，如多串问题，树上字符串问题。只需记录增量处last值即可

///后缀自动机的pre数组构成的树即为反向字符串的后缀树

struct suffix\_auotmation

{

int son[maxn][26],pre[maxn],step[maxn];

int last,si;

void pushback(int v)

{

step[++si]=v;

pre[si]=-1;

clr(son[si],-1);

}

int Extend(int ch)

{

pushback(step[last]+1);

int p=last,np=si;

while(p!=-1&&son[p][ch]==-1)

son[p][ch]=np,p=pre[p];

if(p==-1) pre[np]=0;

else

{

int q=son[p][ch];

if(step[q]==step[p]+1) pre[np]=q;

else

{

pushback(step[p]+1);

int nq=si;

memcpy(son[nq],son[q],sizeof son[q]);

pre[nq]=pre[q];

pre[q]=pre[np]=nq;

while(p!=-1&&son[p][ch]==q)

son[p][ch]=nq,p=pre[p];

}

}

return last=np;

}

void init()

{

si=last=0;

clr(son[0],-1);

pre[0]=-1;

step[0]=0;

}

int right[maxn],pos[maxn];

int cnt[maxn] , end[maxn];

void getright(char \*s)

{

clr(cnt,0);

for(int i=0; i<=si; i++)

cnt[step[i]]++;

for(int i=1; i<=si; i++)

cnt[i]+=cnt[i-1];

for(int i=0; i<=si; i++)

pos[--cnt[step[i]]]=i;

int p=0;

for(int i=0; i<=si; i++) right[i]=0;

for(int i=0; s[i]; i++)

{

p=son[p][s[i]-'a'];

right[p]++;

end[i] = p;

}

for(int i=si; i>=0; i--) if(~pre[pos[i]])

right[pre[pos[i]]] += right[pos[i]];

right[0]=0;

}

void solve(char \*s,int len)

{

int p=0;

int l=0;

for(int i=0; s[i]; i++)

{

int c=s[i]-'a';

while( p!=-1 && son[p][c]==-1 )

p=pre[p],l=step[p];

if(p==-1) p=0;

else

{

p=son[p][c];

l++;

///l为s串从i开始，能匹配的最大长度。

}

}

}

int fa[maxn][25];

void initfa(){

for(int i=0;i<=si;i++)

fa[i][0] = pre[i];

for(int j=1;j<25;j++)

for(int i=0;i<=si;i++)

fa[i][j] = fa[i][j-1]==-1?-1:fa[fa[i][j-1]][j-1];

}

///获取字串在sam中的位置O(lg n)，p为字串结尾，len为子串长度

int getpos(int len,int p){

int u = end[p];

for(int j=24;j>=0;j--)

if( fa[u][j] != -1 && step[fa[u][j]] >= len )

u = fa[u][j];

return u;

}

} su;

char s[maxn],ss[maxn];

int main()

{

// freopen("input.txt","r",stdin);

//freopen("out2.txt","w",stdout);

int n;

while(~scanf("%s",s))

{

su.init();

for(int i=0; s[i]; i++)

su.Extend(s[i]-'a');

su.getright(s);

}

return 0;

}

# 数据结构

## 2D\_bit

//区间修改，单点查询

#include <cstdio>

#include <cmath>

#include <cstring>

#include <string>

#include <algorithm>

#include <queue>

#include <map>

#include <set>

#include <vector>

#include <iostream>

using namespace std;

const int M = 1300;

#define ll long long

int S[M][M];

void upp(int x,int y,ll d){

for(int i=x;i<M;i+=i&-i)

for(int j=y;j<M;j+=j&-j)

S[i][j] += d;

}

ll get(int x,int y){

ll r = 0;

for(int i=x;i;i-=i&-i)

for(int j=y;j;j-=j&-j)

r += S[i][j];

return r;

}

void upp(int x1,int y1,int x2,int y2,ll d){

upp(x1,y1,d);

upp(x2+1,y1,-d);

upp(x1,y2+1,-d);

upp(x2+1,y2+1,d);

}

char s[20];

int x1,x2,yy1,y2;

int main()

{

// freopen("in.txt","r",stdin);

int T,n,m;

scanf("%d",&T);

while(T--){

scanf("%d%d",&n,&m);

memset(S,0,sizeof S);

while(m--){

scanf("%s%d%d",s,&x1,&yy1);

if( \*s == 'Q' )

printf("%lld\n",get(x1,yy1)&1);

else{

scanf("%d%d",&x2,&y2);

upp(x1,yy1,x2,y2,1);

}

}puts("");

}

return 0;

}

//区间修改，区间查询

#include <cstdio>

#include <cmath>

#include <cstring>

#include <string>

#include <algorithm>

#include <queue>

#include <map>

#include <set>

#include <vector>

#include <iostream>

using namespace std;

const int M = 1300;

#define ll long long

ll S[4][M][M];

void upp2(int x,int y,int z,ll d){

// printf("x %d y %d z %d d %lld\n",x,y,z,d);

for(int i=x;i<M;i+=i&-i)

for(int j=y;j<M;j+=j&-j)

S[z][i][j] += d;

}

ll get2(int x,int y,int z){

ll r = 0;

for(int i=x;i;i-=i&-i)

for(int j=y;j;j-=j&-j)

r += S[z][i][j];

// printf(" x %d y %d z %d r %lld\n",x,y,z,r);

return r;

}

///0 s,1 sx,2 sy,3 sxy

void upp1(int x,int y,ll d){

upp2(x,y,0,(x-1)\*(y-1)\*d);

upp2(x,y,1,-(y-1)\*d);

upp2(x,y,2,-(x-1)\*d);

upp2(x,y,3,d);

}

///查询范围是x1 <= x <=x2 , y1 <= y <= y2

void upp(int x1,int y1,int x2,int y2,ll d){

upp1(x1,y1,d);

upp1(x1,y2+1,-d);

upp1(x2+1,y1,-d);

upp1(x2+1,y2+1,d);

}

ll get1(int x,int y){

return get2(x,y,0) + x\*get2(x,y,1) +

y\*get2(x,y,2) + x\*y\*get2(x,y,3);

}

ll get(int x1,int y1,int x2,int y2){

// return get1(x1,y1);

return get1(x1-1,y1-1) - get1(x1-1,y2) -

get1(x2,y1-1) + get1(x2,y2);

}

char s[20];

int x1,x2,yy1,y2;

int main()

{

// freopen("in.txt","r",stdin);

int T,n,m;

scanf("%d",&T);

while(T--){

scanf("%d%d",&n,&m);

memset(S,0,sizeof S);

while(m--){

scanf("%s%d%d",s,&x1,&yy1);

if( \*s == 'Q' )

printf("%lld\n",get(x1,yy1,x1,yy1)&1);

else{

scanf("%d%d",&x2,&y2);

upp(x1,yy1,x2,y2,1);

}

}

}

return 0;

}

## CDQ分治

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

CDQ分治，即中序遍历

可解决降维问题，和动态转静态问题

顺序一般为 (l,m) -> (l,r) -> (m+1,r)

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

HDU 5432 Boring Class

序列 L0～Ln，R0～Rn

选取一些位置，要求L递增，R递减。

输出字典序最小的答案。

解法，因为需要输出字典序最小，需要从后向前进行CDQ

#include<iostream>

#include<cstdio>

#include<algorithm>

#include<cmath>

#include<cstring>

#include <stdio.h>

#include <string>

#define clr(x, y) memset(x, y, sizeof x)

#define inf 1e9

using namespace std;

template<class T>

inline bool read(T &n)

{

T x = 0, tmp = 1;

char c = getchar();

while((c < '0' || c > '9') && c != '-' && c != EOF) c = getchar();

if(c == EOF) return false;

if(c == '-') c = getchar(), tmp = -1;

while(c >= '0' && c <= '9') x \*= 10, x += (c - '0'),c = getchar();

n = x\*tmp;

return true;

}

template <class T>

inline void write(T n)

{

if(n < 0)

{

putchar('-');

n = -n;

}

int len = 0,data[20];

while(n)

{

data[len++] = n%10;

n /= 10;

}

if(!len) data[len++] = 0;

while(len--) putchar(data[len]+48);

}

const int maxn = 55005;

struct node

{

int Li, Ri, id;

bool operator < (const node & A) const

{

return Ri > A.Ri;

}

} q[maxn], tmp[maxn];

int Li[maxn], Ri[maxn], su[maxn];

int dp[maxn], rmax[maxn];

int n;

void upp(int x, int a)

{

while(x <= n)

{

rmax[x] = max(rmax[x], a);

x += x & -x;

}

}

void cl(int x) /// CDQ分治中用到树状数组要这样清空

{

while(x <= n)

{

rmax[x] = 0;

x += x & -x;

}

}

int get\_max(int x)

{

int ans = 0;

while(x > 0)

{

ans = max(ans, rmax[x]);

x -= x & -x;

}

return ans;

}

void CDQ(int L, int R)

{

if (L == R)

{

int &ret = dp[q[L].id];

ret = max(ret, 1);

return ;

}

int mid = (L+R)>>1;

CDQ(mid+1, R);

int L1 = L, L2 = mid+1;

int ptr = mid+1;

sort(q + L, q + mid + 1);

sort(q + mid + 1, q + R+1);

for (int i = L; i <= mid; i++)

{

while(ptr <= R && q[i].Ri <= q[ptr].Ri)

{

upp(q[ptr].Li, dp[q[ptr].id]);

ptr++;

}

int &ret = dp[q[i].id];

ret = max(ret, get\_max(q[i].Li) + 1);

}

for(int i=mid+1;i<=R;i++) ///\*\*特殊清空

cl(q[i].Li);

for (int i = L; i <= mid; i++)

{

while(q[i].id!=i)

swap(q[i], q[q[i].id]);

}

CDQ(L, mid);

}

int main()

{

//freopen("input.txt","r",stdin);

//freopen("output.txt", "w", stdout);

int len;

string ans;

int maxlen, rr;

while(read(n))

{

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

{

read(q[i].Li);

su[i] = q[i].Li;

q[i].id = i;

Li[i] = q[i].Li;

}

sort(su, su + n);

len = unique(su, su + n) - su;

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

q[i].Li = lower\_bound(su, su + len,q[i].Li) - su + 1;

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

{

read(q[i].Ri);

su[i] = q[i].Ri;

Ri[i] = q[i].Ri;

}

clr(dp, 0);

clr(rmax,0);

CDQ(0, n-1);

//for (int i = 0; i < n; i++) printf("Ri=%d i=%d dp=%d\n", q[i].Ri, q[i].id, dp[q[i].id]);

maxlen = 0; rr = -1; ans = "";

for (int i = 0; i < n; i++) maxlen = max(maxlen, dp[i]);

write(maxlen); putchar('\n');

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

{

if (dp[i] == maxlen)

{

if (rr == -1 || (Li[rr] >= Li[i] && Ri[rr] <= Ri[i]))

{

maxlen--;

if(rr!=-1) putchar(' ');

write(i+1);

rr = i;

}

}

}

putchar('\n');

///cout<<"maxlen:"<<maxlen<<endl;

}

return 0;

}

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

hdu 5354 Bipartite Graph

给一张图 ， 要求找到一个点，删除后成为二分图

解法 ： CDQ + 可撤销(种族)并查集

保证在进入（l,r）前，（l，r）内的边没有加入，之外的边全部加入

再递归处理

#include <iostream>

#include <cstdio>

#include <cstdlib>

#include <algorithm>

#include <cmath>

#include <cstring>

#include <vector>

#include <queue>

#include <stack>

#include <set>

#pragma comment(linker, "/STACK:1024000000,1024000000")

#define lson l, m, rt<<1

#define rson m+1, r, rt<<1|1

#define inf 1e9

#define clr(x, y) memset(x, y, sizeof x)

using namespace std;

const int maxn = 200005;

struct edge

{

int to, next;

}G[maxn<<1];

struct node

{

int u, v, hu, hv, fau, fav,colu,colv;

node(){}

node(int a, int b, int c, int d, int e, int f,int g,int h)

{

u = a; v = b; hu = c; hv = d; fau = e; fav = f;

colu=g;

colv=h;

}

};

stack<node> stk;

int head[maxn], si;

int h[maxn], fa[maxn];

int ans[maxn], n, m;

int col[maxn]; //0 代表相同 1 代表不同

void init(int \_n)

{

for (int i = 0; i <= \_n; i++)

{

fa[i] = i;

h[i] = 1;

///

col[i]=0;

}

}

void add(int u, int v)

{

G[si].to = v;

G[si].next = head[u];

head[u] = si++;

}

int find\_fa(int x)

{

int o = x;

while(fa[o] != o) o = fa[o];

return o;

}

int find\_col(int x)

{

if (fa[x] == x) return 1;

///return 1^find\_col(fa[x]);

return col[x]^find\_col(fa[x]);

}

bool Merge(int u, int v)

{

int a = find\_fa(u), b = find\_fa(v);

int x = find\_col(u), y = find\_col(v);

if (a == b)

{

if (x == y)

{

//printf("~~%d %d~fu = %d fv = %d\n", u, v, a, b);

return false;

}

return true;

}

stk.push(node(a, b, h[a], h[b], fa[a], fa[b],col[a],col[b]));

if (h[a] > h[b])

{

fa[b] = a, h[a] += h[b];

///

col[b] = x ^ y ^ 1;

}

else

{

fa[a] = b, h[b] += h[a];

///

col[a] = x ^ y ^ 1;

}

return true;

}

bool unite(int L, int R, int a, int b)

{

for (int u = L; u <= R; u++)

for (int i = head[u]; i != -1; i = G[i].next)

{

int v = G[i].to;

if (a <= v && v <= b) continue;

if (!Merge(u, v)) return false;

}

return true;

}

void get\_del(int x)

{

node tmp;

while(stk.size() > x)

{

tmp = stk.top(); stk.pop();

int u = tmp.u, v = tmp.v;

h[u] = tmp.hu; h[v] = tmp.hv;

fa[u] = tmp.fau; fa[v] = tmp.fav;

/\*\*/col[u] = tmp.colu; col[v] = tmp.colv;

}

}

void cdq(int l,int r)

{

if(l == r)

{

ans[l] = 1;

return ;

}

int pre = stk.size();

int m = (l+r)>>1;

if(unite(m+1, r, l, m)) /// 加入右面的边

cdq(l,m);

else{

for (int i = l; i <= m; i++) ans[i] = 0;

}

get\_del(pre); /// 删去右面的边

if(unite(l, m, m+1, r)) /// 加入左面的边

cdq(m+1,r);

else{

for (int i = m+1; i <= r; i++) ans[i] = 0;

}

get\_del(pre); /// 删去左面的边

return ;

}

int main()

{

//freopen("input.txt", "r", stdin);

int T;

scanf("%d", &T);

while(T--)

{

scanf("%d%d", &n, &m);

init(n);

clr(head, -1); si = 0;

while(stk.size()) stk.pop();

for (int i = 0, st, ed; i < m; i++)

{

scanf("%d%d", &st, &ed);

add(st, ed); add(ed, st);

}

cdq(1, n);

for (int i = 1; i <= n; i++) printf("%d", ans[i]);

printf("\n");

}

return 0;

}

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

bnu 12753 Arnooks's Defensive Line

插入一些区间，并查询有多少区间包含它

解法：裸CDQ动态转静态

（对于某些CDQ问题，可能可以使用先序遍历或后序遍历，但最好用中序遍历）

#include <iostream>

#include <cstdio>

#include <cstdlib>

#include <algorithm>

#include <cmath>

#include <cstring>

#include <vector>

#include <queue>

#include <stack>

#include <set>

#include <vector>

//#pragma comment(linker, "/STACK:1024000000,1024000000")

#define lson l, m, rt<<1

#define rson m+1, r, rt<<1|1

#define inf 1e9

#define debug(a) cout << #a" = " << (a) << endl;

#define debugarry(a, n) for (int i = 0; i < (n); i++) { cout << #a"[" << i << "] = " << (a)[i] << endl; }

#define clr(x, y) memset(x, y, sizeof x)

using namespace std;

typedef unsigned long long uLL;

const int maxn = 500000 + 30;

int ans[maxn];

struct \_\_sad

{

int l,r;

char c;

int id;

bool operator <(const \_\_sad &a) const{

if(l!=a.l) return l<a.l;

else return r < a.r ;

}

}A[maxn],B[maxn],C[maxn];

int rsum[maxn\*4];

void upp(int x,int add)

{

while( x < maxn \* 4 )

{

rsum[x] += add;

x += x&-x;

}

}

void clear(int x)

{

while( x < maxn \* 4 )

{

rsum[x] = 0;

x += x&-x;

}

}

int get(int x)

{

int ret=0;

while(x)

{

ret += rsum[x];

x -= x&-x;

}

return ret;

}

void CDQ(int l,int r)

{

if(l == r) return ;

int mid = l+r >>1;

int lb = 0 ,lc = 0;

for(int i=l;i<=mid;i++)

{

if(A[i].c=='+') B[ lb++ ] = A[i];

}

for(int i=mid+1;i<=r;i++)

{

if(A[i].c=='?') C[ lc++ ] = A[i];

}

sort( B , B + lb );

sort( C , C + lc );

int sum=0;

int cn=0;

for(int i=0;i<lc;i++)

{

while(cn<lb&&B[cn].l<=C[i].l)

{

upp( B[cn].r , 1 );

sum++;

cn++;

}

int ret = sum - get( C[i].r-1 );

ans[ C[i].id ] += ret;

}

for(int i=0;i<lb;i++)

clear(B[i].r);

CDQ( l , mid );

CDQ( mid+1 , r );

}

int sub[maxn\*4],len;

int main()

{

//freopen("input.txt", "r", stdin);

int n;

while(~scanf("%d",&n))

{

char s[30];

len = 0;

for(int i=1;i<=n;i++)

{

scanf("%s%d%d",s,&A[i].l,&A[i].r);

A[i].c=\*s;

A[i].id=i;

sub[len++] = A[i].l;

sub[len++] = A[i].r;

}

sort(sub,sub+len);

len = unique(sub,sub+len) - sub;

for(int i=1;i<=n;i++)

{

A[i].l = lower\_bound(sub,sub+len,A[i].l) - sub + 1;

A[i].r = lower\_bound(sub,sub+len,A[i].r) - sub + 1;

}

clr(ans,0);

clr(rsum,0);

CDQ(1,n);

for(int i=1;i<=n;i++) if(A[i].c=='?')

printf("%d\n",ans[i]);

}

return 0;

}

## Splay

const int maxn = 100000 +30;

#define keyvalue ch[ch[root][1]][0]

#define ls ch[x][0]

#define rs ch[x][1]

int pre[maxn],ch[maxn][2],sz[maxn];

int root,tot1,tot2,s[maxn];

int key[maxn],n,a[maxn];

/\*VVVVVVV\*/

int mi[maxn] , pos[maxn] , rev[maxn];

/\*^^^^^^^\*/

void travel(int x){

if(!x) return ;

travel(ls);

printf("x %d ls %d rs %d pre %d ,mi %d , pos %d ,key %d , rev %d\n",x,ls,rs,pre[x],mi[x],pos[x],key[x],rev[x]);

travel(rs);

}

/\*VVVVVVV\*/

void updaterev(int x){

if(!x) return ;

swap( ls , rs );

pos[x] = sz[x] - pos[x] + 1;

rev[x] ^= 1;

}

/\*^^^^^^^\*/

void pushup(int x){

sz[x] = sz[ch[x][0]] + sz[ch[x][1]] + 1;

/\*VVVVVVV\*/

mi[x] = key[x]; pos[x] = sz[ls]+1;

if( mi[x] > mi[ls] )

mi[x] = mi[ls] , pos[x] = pos[ls];

if( mi[x] > mi[rs] )

mi[x] = mi[rs] , pos[x] = pos[rs] + sz[ls] + 1;

/\*^^^^^^^\*/

}

void pushdown(int x){

if( rev[x] ){

updaterev(ls);

updaterev(rs);

rev[x] = 0;

}

}

void Newnode(int &x,int fa,int k){

if(tot2) x = s[tot2--];

else x = ++tot1;

pre[x] = fa;

key[x] = k;

ls = rs = 0;

sz[x] = 1;

/\*VVVVVVV\*/

mi[x] = k;pos[x] = 1; rev[x] = 0;

/\*^^^^^^^\*/

}

void build(int &x,int l,int r,int fa){

if( l > r ) return ;

int mid=l+r>>1;

Newnode(x,fa,a[mid]);

build(ls,l,mid-1,x);

build(rs,mid+1,r,x);

pushup(x);

}

void Init(){

root = tot1 = tot2 = 0;

ch[root][0] = ch[root][1] = sz[root] = pre[root] = 0;

/\*VVVVVVV\*/

mi[root] = inf; pos[root] = 1;

/\*^^^^^^^\*/

Newnode(root,0,inf);

Newnode(ch[root][1],root,inf);

build(keyvalue,1,n,ch[root][1]);

pushup(ch[root][1]);

pushup(root);

}

void Rotate(int x,int k){

int y = pre[x];

pushdown(y);

pushdown(x);

ch[y][!k] = ch[x][k];

pre[ch[x][k]] = y;

if(pre[y])

ch[pre[y]][ ch[pre[y]][1]==y ] = x;

pre[x] = pre[y];

ch[x][k] = y;

pre[y] = x;

pushup(y);

}

void splay(int r,int goal){

pushdown(r);

while( pre[r] != goal ){

if( pre[pre[r]] == goal )

Rotate(r,ch[pre[r]][0]==r);

else{

int y = pre[r];

int k = ch[pre[y]][0] == y;

if( ch[y][k] == r ){

Rotate(r,!k);

Rotate(r,k);

}else{

Rotate(y,k);

Rotate(r,k);

}

}

}

pushup(r);

if(!goal) root = r;

}

int Getkth(int r,int k){

pushdown(r);

int t = sz[ch[r][0]]+1;

if( t == k ) return r;

if( t > k )

return Getkth(ch[r][0],k);

return Getkth(ch[r][1],k-t);

}

void Rev(int l,int r){

splay(Getkth(root,l),0);

splay(Getkth(root,r+2),root);

updaterev(keyvalue);

pushup(ch[root][1]);

pushup(root);

}

int queryrev(int l,int r){

splay(Getkth(root,l),0);

splay(Getkth(root,r+2),root);

return pos[keyvalue];

}

void Cut(int l,int r,int k){

if( k % (r-l+1) == 0 ) return ;

int c = r - k%(r-l+1) + 1;

splay(Getkth(root,c),0);

splay(Getkth(root,r+2),root);

int tmp = keyvalue;

keyvalue = 0;

pushup(ch[root][1]);

pushup(root);

splay(Getkth(root,l),0);

splay(Getkth(root,l+1),root);

keyvalue = tmp;

pre[tmp] = ch[root][1];

pushup(ch[root][1]);

pushup(root);

}

void ins(int x,ll p){

splay(Getkth(root,x+1),0);

splay(Getkth(root,x+2),root);

Newnode(keyvalue,ch[root][1],p);

pushup(ch[root][1]);

pushup(root);

}

void ear(int x){

if(!x) return ;

s[++tot2] = x;

ear(ls); ear(rs);

}

void del(int x){

splay(Getkth(root,x),0);

splay(Getkth(root,x+2),root);

ear(keyvalue);

keyvalue = 0;

pushup(ch[root][1]);

pushup(root);

}

void tra(int x){

if(!x) return ;

pushdown(x);

tra(ls);

printf("%lld ",key[x]);

tra(rs);

pushup(x);

}

pair<int,int>A[maxn];

int main(){

// freopen("input.txt","r",stdin);

// freopen("out.txt","w",stdout);

while(~scanf("%d",&n)&&n){

for(int i=1;i<=n;i++){

scanf("%d",&a[i]);

A[i] = make\_pair(a[i],i);

}

sort(A+1,A+n+1);

for(int i=1;i<=n;i++)

a[A[i].second] = i;

Init();

for(int i=1;i<=n;i++){

int t = queryrev(i,n) + i -1;

printf("%d%c",t,i==n?'\n':' ');

Rev(i,t);

}

}

return 0;

}

## AAA树

/\*\*

第一行是N和M，表示有这棵树有N个点M个询问

然后是N-1行，每行x,y表示x-y有一条边

接下去是N行，每行是一个数字，表示每个点的权值

后面一行表示根

接下来是M行

第一个数字是K

K=0 表示子树修改，后面x,y，表示以x为根的子树的点权值改成y

K=1 表示换根，后面x，表示把这棵树的根变成x

K=2 表示链修改，后面x,y,z，表示把这棵树中x-y的路径上点权值改成z

K=3 表示子树询问min，后面x，表示以x为根的子树中点的权值min

K=4 表示子树询问max，后面x，表示以x为根的子树中点的权值max

K=5 表示子树加，后面x,y，表示x为根的子树中点的权值+y

K=6 表示链加，后面x,y,z，表示把这棵树中x-y的路径上点权值改成+z

K=7 表示链询问min，后面x,y，表示把这棵树中x-y的路径上点的min

K=8 表示链询问max，后面x,y，表示把这棵树中x-y的路径上点的max

K=9 表示换父亲，后面x,y，表示把x的父亲换成y，如果y在x子树里不操作。

K=10 表示链询问sum，后面x,y,z，表示表示把这棵树中x-y的路径上点的sum

K=11 表示子树询问sum，后面x，表示以x为根的子树的点权sum

\*\*/

#include<bits/stdc++.h>

#define rep(i,a,n) for(int i=a;i<n;i++)

using namespace std;

const int maxn=1e5+1000;

int getint(){

int res=0,f=1;char c=getchar();

while(!isdigit(c))f=f==-1||c=='-'?-1:1,c=getchar();

while(isdigit(c))res=res\*10+c-'0',c=getchar();

return res\*f;

}

int n,m;

struct info{

int mx,mn,sum,sz;

info(){}

info(int mx,int mn,int sum,int sz):

mx(mx),mn(mn),sum(sum),sz(sz){}

void deb(){printf("sum:%d size:%d",(int)sum,sz);}

};

struct flag{

int mul,add;

flag(){mul=1;}

flag(int mul,int add):

mul(mul),add(add){}

bool empty(){return mul==1&&add==0;}

};

info operator+(const info &a,const flag &b) {

return a.sz?info(a.mx\*b.mul+b.add,a.mn\*b.mul+b.add,a.sum\*b.mul+b.add\*a.sz,a.sz):a;

}

info operator+(const info &a,const info &b) {

return info(max(a.mx,b.mx),min(a.mn,b.mn),a.sum+b.sum,a.sz+b.sz);

}

flag operator+(const flag &a,const flag &b) {

return flag(a.mul\*b.mul,a.add\*b.mul+b.add);

}

struct node{

node \*c[4],\*f;

flag Cha,All;

info cha,sub,all;

bool rev,inr;

int val;

void makerev(){rev^=1;swap(c[0],c[1]);}

void makec(const flag &a){

Cha=Cha+a;cha=cha+a;val=val\*a.mul+a.add;

all=cha+sub;

}

void makes(const flag &a,bool \_=1){

All=All+a;all=all+a;sub=sub+a;

if(\_)makec(a);

}

void rz(){

cha=all=sub=info(-(1<<30),1<<30,0,0);

if(!inr)all=cha=info(val,val,val,1);

rep(i,0,2)if(c[i])cha=cha+c[i]->cha,sub=sub+c[i]->sub;

rep(i,0,4)if(c[i])all=all+c[i]->all;

rep(i,2,4)if(c[i])sub=sub+c[i]->all;

}

void pd(){

if(rev){

if(c[0])c[0]->makerev();

if(c[1])c[1]->makerev();

rev=0;

}

if(!All.empty()){

rep(i,0,4)if(c[i])c[i]->makes(All,i>=2);

All=flag(1,0);

}

if(!Cha.empty()){

rep(i,0,2)if(c[i])c[i]->makec(Cha);

Cha=flag(1,0);

}

}

node \*C(int i){if(c[i])c[i]->pd();return c[i];}

bool d(int ty){return f->c[ty+1]==this;}

int D(){rep(i,0,4)if(f->c[i]==this)return i;}

void sets(node \*x,int d){if(x)x->f=this;c[d]=x;}

bool rt(int ty){

if(ty==0)return !f||(f->c[0]!=this&&f->c[1]!=this);

else return !f||!f->inr||!inr;

}

}nd[maxn\*2],\*cur=nd+maxn,\*pool[maxn],\*\*Cur=pool;

int \_cnt;

node \*newnode(){

\_cnt++;

node \*x=(Cur==pool)?cur++:\*(--Cur);

rep(i,0,4)x->c[i]=0;x->f=0;

x->All=x->Cha=flag(1,0);

x->all=x->cha=info(-(1<<30),(1<<30),0,0);

x->inr=1;x->rev=0;x->val=0;

return x;

}

void dele(node \*x){\*(Cur++)=x;}

void rot(node \*x,int ty){

node \*p=x->f;int d=x->d(ty);

if(!p->f)x->f=0;else p->f->sets(x,p->D());

p->sets(x->c[!d+ty],d+ty);x->sets(p,!d+ty);p->rz();

}

void splay(node \*x,int ty=0){

while(!x->rt(ty)){

if(x->f->rt(ty))rot(x,ty);

else if(x->d(ty)==x->f->d(ty))rot(x->f,ty),rot(x,ty);

else rot(x,ty),rot(x,ty);

}x->rz();

}

void add(node \*u,node \*w){

w->pd();

rep(i,2,4)if(!w->c[i]){w->sets(u,i);return;}

node \*x=newnode(),\*v;

for(v=w;v->c[2]->inr;v=v->C(2));

x->sets(v->c[2],2);x->sets(u,3);

v->sets(x,2);splay(x,2);

}

void del(node \*w){

if(w->f->inr){

w->f->f->sets(w->f->c[5-w->D()],w->f->D());

dele(w->f);splay(w->f->f,2);

}else w->f->sets(0,w->D());

w->f=0;

}

void access(node \*w){

static node \*sta[maxn];

static int top=0;

node \*v=w,\*u;

for(u=w;u;u=u->f)sta[top++]=u;

while(top)sta[--top]->pd();

splay(w);

if(w->c[1])u=w->c[1],w->c[1]=0,add(u,w),w->rz();

while(w->f){

for(u=w->f;u->inr;u=u->f);

splay(u);

if(u->c[1])w->f->sets(u->c[1],w->D()),splay(w->f,2);

else del(w);

u->sets(w,1);

(w=u)->rz();

}splay(v);

}

void makert(node \*x){

access(x);x->makerev();

}

node \*findp(node \*u){

access(u);u=u->C(0);

while(u&&u->c[1])u=u->C(1);

return u;

}

node \*findr(node \*u){for(;u->f;u=u->f);return u;}

node\* cut(node \*u){

node \*v=findp(u);

if(v)access(v),del(u),v->rz();

return v;

}

void link(node \*u,node \*v) {

node\* p=cut(u);

if(findr(u)!=findr(v))p=v;

if(p)access(p),add(u,p),p->rz();

}

int main(){

// freopen("bzoj3153.in","r",stdin);

n=getint();m=getint();

static int \_u[maxn],\_v[maxn];

rep(i,1,n)\_u[i]=getint(),\_v[i]=getint();

rep(i,1,n+1){

nd[i].val=getint();

nd[i].rz();

}

rep(i,1,n)makert(nd+\_u[i]),link(nd+\_u[i],nd+\_v[i]);

int root=getint();

makert(nd+root);

// deb();

int x,y,z;

node \*u,\*v;

while(m--){

int k=getint();x=getint();

u=nd+x;

if(k==0||k==3||k==4||k==5||k==11){

access(u);

if(k==3||k==4||k==11){

int ans=u->val;

rep(i,2,4)if(u->c[i]){

info res=u->c[i]->all;

if(k==3) ans=min(ans,res.mn);

else if(k==4) ans=max(ans,res.mx);

else if(k==11) ans+=res.sum;

}printf("%d\n",ans);

}else{

y=getint();

flag fg(k==5,y);

u->val=u->val\*fg.mul+fg.add;

rep(i,2,4)if(u->c[i])u->c[i]->makes(fg);

u->rz();

}

}else if(k==2||k==6||k==7||k==8||k==10){

y=getint();

makert(u),access(nd+y),splay(u);

if (k==7||k==8||k==10) {

info ans=u->cha;

if (k==7) printf("%d\n",ans.mn);

else if (k==8) printf("%d\n",ans.mx);

else printf("%d\n",ans.sum);

}else u->makec(flag(k==6,getint()));

makert(nd+root);

}else if(k==9)link(u,nd+getint());

else if(k==1)makert(u),root=x;

}

return 0;

}

## st表求区间最大值与和

const int maxn= 100000;

//maxn 即数组大小

int flog[ ( maxn << 1 ) + 10 ] ;

int A[ maxn ];

int dmax[ maxn ][ 30 ];

int n;

void RMQ\_init(int \*A) //RMQ 初始化

{

for(int i=0 ; i<n ; i++) dmax[ i ][0] = A[i];

for(int j = 1; (1<<j) <= n ; j++)

for(int i= 0 ; i + (1<<j) - 1< n ; i++)

dmax[ i ][ j ] = max( dmax[ i ][ j - 1 ] ,

dmax[ i + (1<<( j - 1 )) ][ j - 1 ] );

flog[ 0 ] = -1;

for( int i = 1; i < 2 \* maxn ; i ++) flog[ i ] = flog[ i >> 1 ] + 1;

}

int RMQ(int L,int R) //RMQ 查询

{

int k = flog[ R - L + 1 ];

return max( dmax[L][k] , dmax[R-(1<<k)+1][k] );

}

int dsum[maxn][30];

void st\_sum\_init(int \*A)

{

for(int i=0 ; i<n ; i++) dmax[ i ][0] = A[i];

for(int j = 1; (1<<j) <= n ; j++)

for(int i= 0 ; i + (1<<j) - 1< n ; i++)

dmax[ i ][ j ] = dmax[ i ][ j - 1 ] + dmax[ i + (1<<( j - 1 )) ][ j - 1 ];

flog[ 0 ] = -1;

for( int i = 1; i < 2 \* maxn ; i ++) flog[ i ] = flog[ i >> 1 ] + 1;

}

int st\_sum(int L,int R)

{

if(L>R) return 0;

int k = flog[R-L+1];

return dmax[L][k] + st\_sum(L+(1<<k),R);

}

## 叉姐线段树

const int maxn = 2e5;

int getid(x,y)

{

return x+y|y!=x;

}

#define ls getid(l,l+r>>1)

#define rs getid((l+r>>1)+1,r)

#define lson l,r,ls

#define rson m+1,r,rs

int rm[maxn<<1];

void pushup(int l,int r,int rt)

{

rm[rt]=rm[ls]+rm[rs];

}

void build(int l,int r,int rt)

{

if( l == r )

{

scanf("%d",&rm[rt]);

return ;

}

int m=l+r>>1;

build(lson);

build(rson);

pushup(l,r,rt);

}

void update(int p,int c,int l,int r,int rt){

if(l==r){

rm[rt]=c;

return ;

}

int m=l+r>>1;

if( p <= m ) update(p,c,lson);

else update(p,c,rson);

pushup(rt);

}

int query(int L,int R,int l,int r,int rt){

if( L<=l && r<=R ){

return rm[rt];

}

int m=l+r>>1;

int ret = 0;

if( L<=m ) ret += query(L,R,lson);

if( m<R ) ret += query(L,R,rson);

return ret;

}

## 插头dp

const int maxn = 13;

const int HASH=10007;

const int maxs=1000010;

struct sad{

int h[HASH],next[maxs],si,st[maxs];

ll v[maxs];

void init(){

clr(h,-1);

si=0;

}

void put(int s,ll vv){

for(int i=h[s%HASH];~i;i=next[i]) if(st[i]==s){

v[i] += vv;

return ;

}

st[si] = s;

v[si] = vv;

next[si] = h[s%HASH];

h[s%HASH] = si++;

}

}Q[2];

int n,m;

char mp[maxn][maxn];

int id[maxn];

void cca(int \*a){

clr(id,-1);

id[0] = 0;

int cnt = 0;

for(int i=0;i<=m;i++){

if( id[a[i]] == -1 ) id[a[i]] = ++cnt;

a[i] = id[a[i]];

}

}

int code(int \*a){

cca(a);

int st = 0;

for(int i=0;i<=m;i++){

st <<= 3;

st |= a[i];

}

return st;

}

void decode(int st,int \*a){

for(int i=m;i>=0;i--){

a[i] = st&7;

st >>= 3;

}

}

int \*shift(int \*a){

for(int i=m;i>=1;i--)

a[i] = a[i-1];

a[0] = 0;

return a;

}

int a[maxn],r;

void putblack(int i,int j){

for(int k=0;k<Q[r].si;k++){

int st = Q[r].st[k];

ll v = Q[r].v[k];

decode(st,a);

if( a[j] == 0 && a[j-1] == 0 )

Q[r^1].put( st , v );

}

}

void putblock1(int i,int j){

for(int k=0;k<Q[r].si;k++){

int st = Q[r].st[k];

ll v = Q[r].v[k];

decode(st,a);

if( a[j] == 0 && a[j-1] == 0 ){

a[j] = 9;

Q[r^1].put( code(cca(a)) , v );

swap(a[j],a[j-1]);

Q[r^1].put( code(cca(a)) , v );

}else if( a[j-1] == 0 && a[j] ){

a[j] = 0;

Q[r^1].put( code(cca(a)) , v );

}else if( a[j-1] && a[j] == 0 ){

a[j-1] = 0;

Q[r^1].put( code(cca(a)) , v );

}

}

}void putblock2(int i,int j){

for(int k=0;k<Q[r].si;k++){

int st = Q[r].st[k];

ll v = Q[r].v[k];

decode(st,a);

if( a[j] == 0 && a[j-1] == 0 ){

a[j] = a[j-1] = 9;

Q[r^1].put( code(cca(a)) , v );

}else if( a[j] && a[j-1] ){

if( a[j] != a[j-1] ){

for(int t=0;t<=m;t++)

if( t != j && a[t] == a[j] )

a[t] = a[j-1];

a[j] = a[j-1] = 0;

Q[r^1].put( code(cca(a)) , v );

}

}else{

Q[r^1].put( st , v );

swap(a[j],a[j-1]);

Q[r^1].put( code(a) , v );

}

}

}void putshift(){

for(int k=0;k<Q[r].si;k++){

int st = Q[r].st[k];

ll v = Q[r].v[k];

decode(st,a);

if( a[m] == 0 )

Q[r^1].put( code(shift(a)) , v );

}

}

ll solve(){

if( mp[n][1] != '.' || mp[n][m] != '.' )

return 0;

r = 0;

Q[r].init();

Q[r].put(0,1);

for(int i=1;i<=n;i++){

for(int j=1;j<=m;j++){

Q[r^1].init();

if( mp[i][j] == '#' )

putblack(i,j);

else if( mp[i][j] == '.' )

{

if( i==n && (j==1||j==m) )

putblock1(i,j);

else

putblock2(i,j);

}

r^=1;

}

Q[r^1].init();

putshift();

r^=1;

}

for(int k=0;k<Q[r].si;k++){

int st = Q[r].st[k];

ll v = Q[r].v[k];

decode(st,a);

if( st == 0 ) return v;

}

return 0;

}

## 插头dp(2)

#include <cstdio>

#include <cstring>

#include <iostream>

#include <algorithm>

#include <cmath>

#include <vector>

#include <queue>

#include <stack>

#include <set>

#include <map>

#include <complex>

//#include <tr1/unordered\_set>

#include <tr1/unordered\_map>

#include <bitset>

//#pragma comment(linker, "/STACK:1024000000,1024000000")

#define lson l, m, rt<<1

#define rson m+1, r, rt<<1|1

#define inf 1e9+7

#define debug(a) cout << #a" = " << (a) << endl;

#define debugarry(a, n) for (int i = 0; i < (n); i++) { cout << #a"[" << i << "] = " << (a)[i] << endl; }

#define clr(x, y) memset(x, y, sizeof x)

#define LL long long

#define uLL unsigned LL

using namespace std;

const int maxn = 20;

struct sad

{

int w,l,r,f;

int hash()

{

return (f<<30) + (w<<20) + (l<<10) +r;

}

sad(int h)

{

f = (h>>30);

w = ( (h>>20) & ((1<<10)-1) );

l = ( (h>>10) & ((1<<10)-1) );

r = ( h & ((1<<10)-1) );

}

sad() {};

void print()

{

printf("w %d l %d r %d\n",w,l,r);

}

};

tr1::unordered\_map<int,int>Q[2];

tr1::unordered\_map<int,int>::iterator it;

int n;

int mp[maxn][maxn];

int solve()

{

int r = 0;

Q[0].clear();

Q[1].clear();

Q[0][0]=0;

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

{

for(int j=0; j<n; j++)

{

//printf("i %d j %d\n",i,j);

Q[r^1].clear();

for(it=Q[r].begin(); it!=Q[r].end(); it++)

{

int c = it->second;

sad b = sad(it->first);

//b.print();

//printf("c %d\n",c);

if( mp[i][j] )

{

b.w &= ~(1<<j);

b.l &= ~(1<<j);

b.r &= ~(1<<j);

b.f = 0;

Q[r^1][b.hash()] = max( Q[r^1][b.hash()] , c );

}

else

{

Q[r^1][b.hash()] = max( Q[r^1][b.hash()] , c );

if( (b.w&(1<<j)) || (b.l&(1<<j)) || (b.r&(1<<j)) )

continue;

else

{

if( b.f == 0 )

{

sad e;

e.w = (b.w | (1<<j));

e.l = (b.l | (1<<j));

e.r = (b.r | (1<<j));

e.f = 1;

Q[r^1][e.hash()] = max( Q[r^1][e.hash()] , c + 1 );

}

}

}

}

r^=1;

}

Q[r^1].clear();

for(it=Q[r].begin(); it!=Q[r].end(); it++)

{

int c = it->second;

sad b = sad(it->first);

sad e;

e.w = b.w;

e.l = (b.l<<1) % (1<<n);

e.r = (b.r>>1);

e.f = 0;

Q[r^1][e.hash()] = max( Q[r^1][e.hash()] , c );

}

r^=1;

}

int ans = 0;

for(it=Q[r].begin(); it!=Q[r].end(); it++)

ans = max( ans , it->second );

return ans;

}

int main()

{

//freopen("input.txt", "r", stdin);

//freopen("out1","w",stdout);

int T;

scanf("%d",&T);

while(T--)

{

scanf("%d",&n);

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

for(int j=0; j<n; j++)

scanf("%d",&mp[i][j]);

printf("%d\n",solve());

}

return 0;

}

## 莫队算法

#include <cstdio>

#include <cstring>

#include <iostream>

#include <algorithm>

#include <cmath>

#include <vector>

#include <queue>

#include <stack>

#include <set>

#include <map>

#include <complex>

//#include <tr1/unordered\_set>

//#include <tr1/unordered\_map>

#include <bitset>

//#pragma comment(linker, "/STACK:1024000000,1024000000")

#define lson l, m, rt<<1

#define rson m+1, r, rt<<1|1

#define inf 1e18+7

#define debug(a) cout << #a" = " << (a) << endl;

#define debugarry(a, n) for (int i = 0; i < (n); i++) { cout << #a"[" << i << "] = " << (a)[i] << endl; }

#define clr(x, y) memset(x, y, sizeof x)

#define LL long long

#define uLL unsigned LL

using namespace std;

const int maxn = 30000+30;

int block;

struct sad

{

int l,r;

sad(){};

sad(int ll,int rr){l=ll;r=rr;};

bool operator<(const sad &b)const{

if(l/block==b.l/block)

return r<b.r;

return l<b.l;

}

};

int Hash(int l,int r){

return l\*maxn +r;

}

vector<sad>Q;

map<int,int>G;

int K,n;

int sum[maxn] , val[maxn];

int ans;

void chen(int p,int d)

{

int t = K-val[p];

if(t>=1&&t<=n)

{

ans -= sum[val[p]] \*sum[t];

sum[val[p]]+=d;

ans += sum[val[p]] \*sum[t];

}

else

sum[val[p]]+=d;

}

void solve()

{

clr(sum,0);

ans=0;

int l=1,r=0;

block = sqrt(n)+1;

sort(Q.begin(),Q.end());

for(int i=0;i<Q.size();i++)

{

while(l<Q[i].l)

{

chen(l,-1);

l++;

}

while(l>Q[i].l)

{

l--;

chen(l,1);

}

while(r<Q[i].r)

{

r++;

chen(r,1);

}

while(r>Q[i].r)

{

chen(r,-1);

r--;

}

G[ Hash(Q[i].l,Q[i].r) ] = ans;

}

}

int l[maxn] , r[maxn] , u[maxn] , v[maxn];

int cnt[maxn];

int main()

{

//freopen("input.txt", "r", stdin);

while(~scanf("%d%d",&n,&K))

{

cnt[0]=0;

for(int i=1;i<=n;i++)

{

scanf("%d",&val[i]);

cnt[i]=cnt[i-1];

if(val[i]\*2==K)

cnt[i]++;

}

Q.clear();

G.clear();

int m;

scanf("%d",&m);

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

{

scanf("%d%d%d%d",&l[i],&r[i],&u[i],&v[i]);

Q.push\_back(sad(l[i],v[i]));

Q.push\_back(sad(l[i],u[i]-1));

Q.push\_back(sad(r[i]+1,v[i]));

Q.push\_back(sad(r[i]+1,u[i]-1));

}

solve();

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

{

int ans=0;

ans = G[Hash(l[i],v[i])] - G[Hash(l[i],u[i]-1)]

- G[Hash(r[i]+1,v[i])] + G[Hash(r[i]+1,u[i]-1)];

// printf("%d %d %d %d\n",G[Hash(l[i],v[i])] , G[Hash(l[i],u[i]-1)]

// , G[Hash(r[i]+1,v[i])] , G[Hash(r[i]+1,u[i]-1)]);

ans -= (cnt[r[i]] - cnt[l[i]-1]) \* (cnt[v[i]] - cnt[u[i]-1]);

printf("%d\n",ans);

}

}

return 0;

}

## 带修改莫队

/\*\*

墨墨购买了一套N支彩色画笔（其中有些颜色可能相同），摆成一排，你需要回答墨墨的提问。墨墨会像你发布如下指令： 1、 Q L R代表询问你从第L支画笔到第R支画笔中共有几种不同颜色的画笔。 2、 R P Col 把第P支画笔替换为颜色Col。为了满足墨墨的要求，你知道你需要干什么了吗？

Input

第1行两个整数N，M，分别代表初始画笔的数量以及墨墨会做的事情的个数。第2行N个整数，分别代表初始画笔排中第i支画笔的颜色。第3行到第2+M行，每行分别代表墨墨会做的一件事情，格式见题干部分。

Output

对于每一个Query的询问，你需要在对应的行中给出一个数字，代表第L支画笔到第R支画笔中共有几种不同颜色的画笔。

对于100%的数据，N≤10000，M≤10000，修改操作不多于1000次，所有的输入数据中出现的所有整数均大于等于1且不超过10^6

\*\*/

#include<iostream>

#include<cstring>

#include<cstdio>

#include<algorithm>

#include<cmath>

#include<assert.h>

using namespace std;

#define ll long long

const int maxn = 20000 + 200;

int block;

struct sad{

int l,r,ti,id;

bool operator<(const sad &B)const{

if( l/block != B.l/block )

return l < B.l;

if( r/block != B.r/block )

return r < B.r;

return ti < B.ti;

}

sad(){};

sad(int a,int b,int c,int d){

l=a;r=b;ti=c;id=d;

}

}p[maxn],q[maxn];

int ficol[maxn],col[maxn],cntcol[maxn],Ans;

int ans[maxn];

inline void change(int x,int d){

if(d==1){

if( cntcol[x]++ == 0 ) Ans++;

}else{

if( --cntcol[x] == 0 ) Ans--;

}

}

void solve(int n,int m){

sort(p,p+n);

for(int L=0,R;L<n;L=R){

for(R=L;R<n;R++)

if( p[L].l/block != p[R].l/block ) break;

else if( p[L].r/block != p[R].r/block ) break;

// printf("L %d R %d\n",L,R);

memcpy(col,ficol,sizeof col);

memset(cntcol,0,sizeof cntcol);

Ans = 0;

int l = 1 , r = 0 , pos = 0;

for(int i=L;i<R;i++){

// printf("l %d r %d ti %d\n",p[i].l,p[i].r,p[i].ti);

while( pos < m && q[pos].ti < p[i].ti ){

if( q[pos].l>=l && q[pos].l<=r )

change(col[q[pos].l],-1) ,

change(q[pos].r,1);

col[q[pos].l]=q[pos].r;

pos++;

}

while( l < p[i].l ) change(col[l++],-1);

while( l > p[i].l ) change(col[--l],1);

while( r > p[i].r ) change(col[r--],-1);

while( r < p[i].r ) change(col[++r],1);

// cout<<Ans<<endl;

ans[p[i].id] = Ans;

}

}

}

char s[maxn];

int sub[maxn] , len;

int main(){

// freopen("in.txt","r",stdin);

int n,m,m1,m2,a,b;

while(~scanf("%d%d",&n,&m)){

m1 = m2 = 0; len= 0;

block = 1+pow(n+m,0.667);

// printf("block %d\n",block);

for(int i=1;i<=n;i++){

scanf("%d",&ficol[i]);

sub[len++] = ficol[i];

}

for(int i=0;i<m;i++){

scanf("%s%d%d",s,&a,&b);

if(\*s=='Q') p[m1] = sad(a,b,i,m1),m1++;

else q[m2++] = sad(a,b,i,-1),sub[len++] = b;

}

sort(sub,sub+len);

len = unique(sub,sub+len)-sub;

for(int i=1;i<=n;i++)

ficol[i] = lower\_bound(sub,sub+len,ficol[i]) - sub;

for(int i=0;i<m2;i++)

q[i].r = lower\_bound(sub,sub+len,q[i].r) - sub;

solve(m1,m2);

for(int i=0;i<m1;i++)

printf("%d\n",ans[i]);

}

return 0;

}

## 整体二分

#include <iostream>

#include <cstdio>

#include <cstdlib>

#include <algorithm>

#include <cmath>

#include <cstring>

#include <vector>

#include <queue>

#include <stack>

#include <set>

#include <vector>

#include <deque>

#include <set>

//#pragma comment(linker, "/STACK:1024000000,1024000000")

#define lson l, m, ls[rt]

#define rson m+1, r, rs[rt]

#define inf 1e9

#define debug(a) cout << #a" = " << (a) << endl;

#define debugarry(a, n) for (int i = 0; i < (n); i++) { cout << #a"[" << i << "] = " << (a)[i] << endl; }

#define clr(x, y) memset(x, y, sizeof x)

#define LL long long

using namespace std;

const int maxn = 2e6;

struct \_\_sad

{

int l , r , id;

int st , k , v , add;

}p[maxn],p1[maxn],p2[maxn];

int ans[maxn];

int n;

int rsum[maxn];

void upp(int x,int add)

{

while(x<=n)

{

rsum[x] += add;

x += x&-x;

}

}

int get(int x)

{

int ret=0;

while(x)

{

ret += rsum[x];

x -= x&-x;

}

return ret;

}

void Bin(int st,int ed,int l,int r)

{

if(st>ed) return ;

if(l==r)

{

for(int i=st;i<=ed;i++) if( p[i].st==2 )

ans[ p[i].id ] = l;

return ;

}

int mid = l+(r-l)/2;

int ta1=0,ta2=0;

for(int i=st;i<=ed;i++)

{

if(p[i].st==1)

{

if( p[i].v<=mid )

{

p1[ta1++] = p[i];

upp( p[i].l , p[i].add );

}

else p2[ta2++] = p[i];

}

else

{

int t = get( p[i].r ) - get( p[i].l - 1 );

if( t >= p[i].k ) p1[ta1++]=p[i];

else p[i].k -= t , p2[ta2++]=p[i];

}

}

for(int i=st;i<=ed;i++)

{

if(p[i].st==1)

if( p[i].v<=mid )

upp( p[i].l , p[i].add\*(-1) );

}

for(int i=0;i<ta1;i++)

p[ i+st ] = p1[ i ];

for(int i=0;i<ta2;i++)

p[ i+st+ta1 ] = p2[ i ];

Bin( st , st + ta1 - 1 , l , mid );

Bin( st + ta1 , ed , mid+1 , r );

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

整体二分如果用于处理边查询边修改的问题，下标本身就是一维，不能在Bin子区间之前加入东西

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

int A[maxn];

int sub[maxn],len;

int main()

{

//freopen("1007.in", "r", stdin);

//freopen("output.txt","w",stdout);

int q;

while(~scanf("%d",&n))

{

int cnt=0,acnt=0;

int a,st,l,r,v,k;

len=0;

for(int i=1;i<=n;i++)

{

scanf("%d",&a);

A[i]=a;

sub[len++]=a;

p[cnt].l=i; p[cnt].st=1;

p[cnt].v=a; p[cnt].add=1;

cnt++;

}

scanf("%d",&q);

for(int i=0;i<q;i++)

{

scanf("%d",&st);

if(st==1)

{

scanf("%d%d",&l,&v);

sub[len++]=v;

p[cnt].l=l; p[cnt].st=1;

p[cnt].v=A[l]; p[cnt].add=-1;

cnt++;

p[cnt].l=l; p[cnt].st=1;

p[cnt].v=A[l]=v; p[cnt].add=1;

cnt++;

}

else{

scanf("%d%d%d",&l,&r,&k);

p[cnt].l=l; p[cnt].r=r;

p[cnt].st=2; p[cnt].k=k;

p[cnt].id=acnt++;

cnt++;

}

}

sort(sub,sub+len);

len = unique( sub , sub + len ) - sub;

for(int i=0;i<cnt;i++) if(p[i].st==1)

{

p[i].v = lower\_bound( sub , sub+len , p[i].v ) - sub + 1;

}

Bin(0,cnt-1,0,len+7);

for(int i=0;i<acnt;i++)

printf( "%d\n" , sub[ ans[i] - 1 ] );

}

return 0;

}

Description

There are shops numbered with successive integers from to in Byteland. Every shop sells only one kind of goods, and the price of the -th shop's goods is . The distance between the -th shop and Byteasar's home is .

Every day, Byteasar will purchase some goods. On the -th day, he will choose an interval and an upper limit . Then, he will visit each shop with distance at most away from home in , buy at most one piece of goods from each shop and go back home. Of course, he can also choose to buy nothing. Back home, Byteasar will calculate the total amount of money he has costed that day and write it down on his account book, denoted as .

However, due to Byteasar's poor math, he may calculate it wrong.

Please write a program to help Byteasar judge whether each number is sure to be calculated wrong.

Input

The first line of the input contains an integer , denoting the number of test cases.

In each test case, the first line of the input contains two integers , denoting the number of shops and the number of records on Byteasar's account book.

The second line of the input contains integers , denoting the price of the -th shop's goods.

The third line of the input contains integers , denoting the distance between the -th shop and Byteasar's home.

Each of the next lines contains four integers , denoting a record on Byteasar's account book.

Output

For each test case, print a line with characters. If the -th number is sure to be calculated wrong, then the -th character should be '1'. Otherwise, it should be '0'.

Sample Input

2

3 3

3 3 3

2 4 3

3 3 5 3

3 3 3 1

2 3 1 3

5 4

5 1 2 4 2

1 8 9 2 1

1 5 1 3

4 4 1 5

1 5 3 5

1 3 5 1

Sample Output

011

1101

#include <stdio.h>

#include <string.h>

#include <iostream>

#include <vector>

#include <algorithm>

#include <bitset>

#define lson l,m,rt<<1

#define rson m+1,r,rt<<1|1

using namespace std;

const int maxn = 2e5;

const int inf = 1e9+7;

struct sad{

int l,r,c,sum,id;

}q[maxn];

int v[maxn] , d[maxn];

int ans[maxn];

int f[30000][120] , M = 100;

void CDQ(int l,int r,int st,int ed){

if( st > ed ) return ;

int mid=l+r>>1,w1=st-1,w2=ed;

for(int i=st;i<=w2;i++)

if( q[i].r < mid )

swap(q[i],q[++w1]);

else if( q[i].l > mid )

swap(q[i],q[w2--]) , i--;

CDQ(l,mid,st,w1);

CDQ(mid+1,r,w2+1,ed);

/// st w1 , w1+1 w2 , w2+1 ed

for(int i=l;i<=r;i++)

for(int j=0;j<=M;j++)

f[i][j] = inf;

f[mid][0] = 0;

for(int i=mid-1;i>=l;i--)

for(int j=0;j<=M;j++)

{

f[i][j] = f[i+1][j];

if(j>=v[i])

f[i][j] = min(f[i][j],max(f[i+1][j-v[i]],d[i]) );

}

f[mid][v[mid]] = d[mid];

for(int i=mid+1;i<=r;i++)

for(int j=0;j<=M;j++)

{

f[i][j] = f[i-1][j];

if(j>=v[i])

f[i][j] = min(f[i][j],max(f[i-1][j-v[i]],d[i]) );

}

for(int i=w1+1;i<=w2;i++){

ans[q[i].id] = 1;

for(int j=0;j<=q[i].sum;j++){

if( q[i].l == mid && j != 0 )

continue;

if( max(f[q[i].l][j],f[q[i].r][q[i].sum-j]) <= q[i].c ){

ans[q[i].id] = 0;

break;

}

}

}

}

int main()

{

// freopen("in.txt","r",stdin);

int T,n,m;

scanf("%d",&T);

while(T--){

scanf("%d%d",&n,&m);

for(int i=1;i<=n;i++)

scanf("%d",&v[i]);

for(int i=1;i<=n;i++)

scanf("%d",&d[i]);

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

scanf("%d%d%d%d",&q[i].l,&q[i].r,

&q[i].c,&q[i].sum),q[i].id = i;

CDQ(1,n,0,m-1);

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

printf("%d",ans[i]);

puts("");

}

return 0;

}

## 静态主席树

#include <iostream>

#include <cstdio>

#include <cstdlib>

#include <algorithm>

#include <cmath>

#include <cstring>

#include <vector>

#include <queue>

#include <stack>

#include <set>

#include <vector>

#include <deque>

#include <set>

//#pragma comment(linker, "/STACK:1024000000,1024000000")

#define lson l, (l+r>>1), ls[rt]

#define rson (l+r>>1)+1, r, rs[rt]

#define inf 1e9

#define debug(a) cout << #a" = " << (a) << endl;

#define debugarry(a, n) for (int i = 0; i < (n); i++) { cout << #a"[" << i << "] = " << (a)[i] << endl; }

#define clr(x, y) memset(x, y, sizeof x)

#define LL long long

using namespace std;

const int maxn = 1e5+20,maxs=maxn\*20;

#define head(p) ( p >= 0 ? h[p] : 0 )

struct \_\_sad

{

int ls[maxs],rs[maxs];

int sum[maxs];

int h[maxn];

int si,len;

void pushup(int rt)

{

sum[rt] = sum[ls[rt]] + sum[rs[rt]];

}

void build(int p,int add,int l,int r,int &rt,int rt2)

{

if(!rt)

{

if(p<l||p>r)

{

rt=rt2;

return ;

}

sum[si]=ls[si]=rs[si]=0;

rt=si++;

}

if(l==r)

{

sum[rt] = sum[rt2] + add;

return ;

}

build(p,add,lson,ls[rt2]);

build(p,add,rson,rs[rt2]);

pushup(rt);

}

void init(int \*A,int n)

{

clr(h,0);

si = 1;

ls[0]=rs[0]=sum[0]=0;

len = n;

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

{

build( A[i] , 1 , 0 , n , head(i-1) );

}

}

int query(int k,int l,int r,int rt,int rt2)

{

if(l==r) return l;

int nk = sum[ls[rt]]-sum[ls[rt2]];

if(nk>=k) return query(k,lson,ls[rt2]);

return query(k-nk,rson,rs[rt2]);

}

int query(int L,int R,int k)

{

return query(k,0,len,head(R),head(L-1));

}

};

## 动态主席树

用于解决动态修改某一个数，动态查询区间第k大

空间复杂度为 nlgnlgn

时间复杂度为 nlgnlgn

zoj 2112 Dynamic Rankings

裸动态主席树（其实这道题用整体二分更好）

N 50000 M 10000

输入格式

N M

a1 ... an

(M){

Q l r k

C pos t

}

#include <iostream>

#include <cstdio>

#include <cstdlib>

#include <algorithm>

#include <cmath>

#include <cstring>

#include <vector>

#include <queue>

#include <stack>

#include <set>

#include <vector>

#include <map>

#include <tr1/unordered\_set>

#include <tr1/unordered\_map>

//#pragma comment(linker, "/STACK:1024000000,1024000000")

#define inf 1e9

#define debug(a) cout << #a" = " << (a) << endl;

#define debugarry(a, n) for (int i = 0; i < (n); i++) { cout << #a"[" << i << "] = " << (a)[i] << endl; }

#define clr(x, y) memset(x, y, sizeof x)

#define LL long long

using namespace std;

const int maxn = 100010; /// N×2

const int M = 8000030; /// （M+N)\*800

int n, q, m, tot;

int a[maxn], t[maxn];

int T[maxn], lson[M], rson[M], c[M];

int S[maxn];

struct Query

{

int kind;

int l, r, k;

} query[100010];

void Init\_hash(int k)

{

sort(t, t + k);

m = unique(t, t + k) - t;

}

int get\_hash(int x)

{

return lower\_bound(t, t + m, x) - t;

}

int build(int l, int r)

{

int root = tot++;

c[root] = 0;

if (l != r)

{

int mid = (l+r)>>1;

lson[root] = build(l, mid);

rson[root] = build(mid+1, r);

}

return root;

}

int Insert(int root, int pos, int val)

{

int newroot = tot++, tmp = newroot;

int l = 0, r = m-1;

c[newroot] = c[root] + val;

while(l < r)

{

int mid = (l+r)>>1;

if (pos <= mid)

{

lson[newroot] = tot++; rson[newroot] = rson[root];

newroot = lson[newroot]; root = lson[root];

r = mid;

}

else

{

rson[newroot] = tot++; lson[newroot] = lson[root];

newroot = rson[newroot]; root = rson[root];

l = mid+1;

}

c[newroot] = c[root] + val;

}

return tmp;

}

int lowbit(int x)

{

return x & (-x);

}

int use[maxn];

void add(int x, int pos, int val)

{

while(x <= n)

{

S[x] = Insert(S[x], pos, val);

x += lowbit(x);

}

}

int sum(int x)

{

int ret = 0;

while(x > 0)

{

ret += c[lson[use[x]]];

x -= lowbit(x);

}

return ret;

}

int Query(int left, int right, int k)

{

int left\_root = T[left - 1];

int right\_root = T[right];

int l = 0, r = m-1;

for (int i = left-1; i; i -= lowbit(i)) use[i] = S[i];

for (int i = right; i; i -= lowbit(i)) use[i] = S[i];

while(l < r)

{

int mid = (l+r)/2;

int tmp = sum(right) - sum(left-1) + c[lson[right\_root]] - c[lson[left\_root]];

if (tmp >= k)

{

r = mid;

for (int i = left - 1; i; i -= lowbit(i))

use[i] = lson[use[i]];

for (int i = right; i; i -= lowbit(i))

use[i] = lson[use[i]];

left\_root = lson[left\_root];

right\_root = lson[right\_root];

}

else

{

l = mid+1;

k -= tmp;

for (int i = left-1; i ; i -= lowbit(i) )

use[i] = rson[use[i]];

for(int i = right;i;i-=lowbit(i))

use[i] = rson[use[i]];

left\_root = rson[left\_root];

right\_root = rson[right\_root];

}

}

return l;

}

void Modify(int x,int p,int d)

{

while(x<=n)

{

S[x] = Insert(S[x],p,d);

x += lowbit(x);

}

}

int main()

{

//freopen("input.txt","r",stdin);

while(~scanf("%d",&n))

{

tot = 0;

m = 0;

q = maxn;

for(int i=1;i<=n;i++)

{

scanf("%d",&a[i]);

t[m++] = a[i];

}

scanf("%d",&q);

int op;

for(int i=0;i<q;i++)

{

scanf("%d",&op);

if(op==2)

{

query[i].kind = 0;

scanf("%d%d%d",&query[i].l,&query[i].r,&query[i].k);

}

else

{

query[i].kind=1;

scanf("%d%d",&query[i].l,&query[i].r);

t[m++] = query[i].r;

}

}

Init\_hash(m);

T[0] = build(0,m-1);

for(int i=1;i<=n;i++)

T[i] = Insert( T[i-1] , get\_hash(a[i]) , 1 );

for(int i=1;i<=n;i++)

S[i] = T[0];

for(int i=0;i<q;i++)

{

if(query[i].kind==0)

printf("%d\n",t[Query(query[i].l,query[i].r,query[i].k)]);

else

{

Modify(query[i].l,get\_hash(a[query[i].l]),-1);

Modify(query[i].l,get\_hash(query[i].r),1);

a[query[i].l] = query[i].r;

}

}

}

return 0;

}

## 可持续化并查集

可持续化并查集可以用可持续化线段树维护的可持续化数组来维护

按秩合并的并查集可以保证Find的时间复杂度为O(logn)

bzoj 3673

n个集合 m个操作

操作：

1 a b 合并a,b所在集合

2 k 回到第k次操作之后的状态(查询算作操作)

3 a b 询问a,b是否属于同一集合，是则输出1否则输出0

0<n,m<=2\*10^4

const int maxn=6e5+30;

int si,ls[maxn],rs[maxn];

int fa[maxn],ra[maxn];

int newnode(int la)

{

if(si>maxn-10) while(1);

++si;

ls[si]=ls[la];

rs[si]=rs[la];

fa[si]=fa[la];

ra[si]=ra[la];

return si;

}

void build(int l,int r,int rt)

{

if(l==r)

{

fa[rt]=l;

ra[rt]=0;

return ;

}

int m=l+r>>1;

ls[rt]=++si;

rs[rt]=++si;

build(lson);

build(rson);

}

int query(int pos,int l,int r,int rt)

{

if(l==r) return rt;

int m=l+r>>1;

if( pos <= m ) return query(pos,lson);

return query(pos,rson);

}

void update(int pos,int ffa,int rra,int l,int r,int rt)

{

if(l==r)

{

fa[rt]=ffa;

ra[rt]=rra;

return ;

}

int m=l+r>>1;

if( pos <= m )

{

ls[rt]=newnode(ls[rt]);

update(pos,ffa,rra,lson);

}else{

rs[rt]=newnode(rs[rt]);

update(pos,ffa,rra,rson);

}

}

int h[maxn],n;

int Find(int x,int t)

{

int f = fa[ query(x,1,n,h[t]) ];

if( f==x ) return x;

return Find(f,t);

}

int main()

{

// freopen("input.txt","r",stdin);

//freopen("out2.txt","w",stdout);

int m;

while(~scanf("%d%d",&n,&m))

{

h[0]=si=0;

build(1,n,h[0]);

for(int i=1;i<=m;i++)

{

h[i]=newnode(h[i-1]);

int p,u,v,k;

scanf("%d",&p);

if( p == 1 )

{

scanf("%d%d",&u,&v);

int x = Find(u,i) , y = Find(v,i);

if( x != y )

{

int rax=ra[query(x,1,n,h[i])] ,

ray=ra[query(y,1,n,h[i])];

if( rax > ray )

swap(x,y),swap(rax,ray);

update(x,y,rax,1,n,h[i]);

if( rax == ray )

update(y,y,ray+1,1,n,h[i]);

}

}

else if( p == 2 )

{

scanf("%d",&k);

h[i]=newnode(h[k]);

}else{

scanf("%d%d",&u,&v);

int x = Find(u,i) , y = Find(v,i);

printf("%d\n",x==y);

}

}

}

return 0;

}

## 多重背包

int dp[maxn];

int w[maxn],v[maxn],c[maxn];

int n,C;

void f()

{

clr(dp,0);

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

{

int cnt = c[i];

int k = 1;

while(cnt>=k)

{

for(int j=C;j>=w[i]\*k;j--)

{

dp[j] = max(dp[j],dp[j-w[i]\*k]+v[i]\*k);

}

cnt-=k;

k<<=1;

}

if(cnt)

{

k=cnt;

for(int j=C;j>=w[i]\*k;j--)

{

dp[j] = max(dp[j],dp[j-w[i]\*k]+v[i]\*k);

}

}

}

}

## 二维st表

#include <cstdio>

#include <iostream>

#include <algorithm>

#include <set>

#include <map>

#include <vector>

#include <deque>

#include <string.h>

using namespace std;

const int maxn = 1010;

int flog[maxn\*2+100];

int dmax[maxn][maxn][11][11];

int n,m;

int a[maxn][maxn];

void RMQ\_init(){

for(int k1=0;(1<<k1)<=n;k1++)

for(int k2=0;(1<<k2)<=m;k2++)

for(int i=0;i+(1<<k1)<=n;i++)

for(int j=0;j+(1<<k2)<=m;j++){

if( k1 )

dmax[i][j][k1][k2] = max(dmax[i][j][k1-1][k2],

dmax[i+(1<<(k1-1))][j][k1-1][k2]);

else if( k2 )

dmax[i][j][k1][k2] = max(dmax[i][j][k1][k2-1],

dmax[i][j+(1<<(k2-1))][k1][k2-1]);

else

dmax[i][j][k1][k2] = a[i][j];

}

flog[0] = -1;

for(int i=1;i<2\*maxn;i++)

flog[i] = flog[i>>1]+1;

}

int RMQ(int x1,int y1,int x2,int y2){

int k1 = flog[x2-x1+1] , k2 = flog[y2-y1+1];

int x3 = x2-(1<<k1)+1 , y3 = y2-(1<<k2)+1;

return max( dmax[x1][y1][k1][k2] ,

max(dmax[x1][y3][k1][k2],max(

dmax[x3][y1][k1][k2] , dmax[x3][y3][k1][k2] ) ) );

}

int mp[maxn][maxn] , p[3][maxn][maxn];

int main()

{

// freopen("in.txt","r",stdin);

// printf("size %d\n",(sizeof dmax+sizeof a)>>20);

while(~scanf("%d%d ",&n,&m)){

memset(p,0,sizeof p);

memset(a,0,sizeof a);

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

for(int j=0;j<m;j++)

mp[i][j]=getchar()-'0' , getchar();

for(int i=n-1;i>=0;i--)

for(int j=m-1;j>=0;j--) if(mp[i][j]){

p[0][i][j] = p[0][i+1][j]+1;

p[1][i][j] = p[1][i][j+1]+1;

p[2][i][j] = min(p[2][i+1][j+1]+1,

min(p[0][i][j],p[1][i][j]));

a[i][j] = p[2][i][j];

}

RMQ\_init();

int q,x1,x2,y1,y2;

scanf("%d",&q);

while(q--){

scanf("%d%d%d%d",&x1,&y1,&x2,&y2);

x1--;

x2--;

y1--;

y2--;

int l = 1 , r = min(x2-x1+1,y2-y1+1) , ans = 0;

while(l<=r){

int m=l+r>>1;

if( RMQ(x1,y1,x2-m+1,y2-m+1) >= m ){

ans = max(ans,m);

l = m+1;

}else r = m-1;

}

printf("%d\n",ans);

}

}

return 0;

}

## 康拓展开

#include <cstdio>

#include <cstring>

#include <iostream>

#include <algorithm>

#include <cmath>

#include <vector>

#include <queue>

#include <stack>

#include <set>

#include <map>

#include <complex>

//#include <tr1/unordered\_set>

//#include <tr1/unordered\_map>

#include <bitset>

//#pragma comment(linker, "/STACK:1024000000,1024000000")

#define lson l, m, rt<<1

#define rson m+1, r, rt<<1|1

#define inf 1e9+7

#define debug(a) cout << #a" = " << (a) << endl;

#define debugarry(a, n) for (int i = 0; i < (n); i++) { cout << #a"[" << i << "] = " << (a)[i] << endl; }

#define clr(x, y) memset(x, y, sizeof x)

#define LL long long

#define uLL unsigned LL

using namespace std;

const int maxn = 9;

int n=9;

int base[maxn];

int Hash(int \*a)

{

int ret=0;

for(int i=n-1;i>=0;i--)

{

int c=0;

for(int j=i+1;j<n;j++)

if(a[j]>a[i])

c++;

ret += c \* base[n-1-i];

}

//debug(ret);

return ret;

}

void get\_hash(int h,int \*a)

{

int us[maxn];

clr(us,0);

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

{

int c = h / base[n-1-i];

h %= base[n-1-i];

int j=n-1;

while(c||us[j])

c -= (1^us[j--]);

a[i] = j;

us[j] = 1;

}

}

int num[maxn] , ans[maxn];

int main()

{

//freopen("input.txt","r",stdin);

//freopen("out1","w",stdout);

base[0] = 1;

for(int i=1;i<n;i++)

base[i] = i \* base[i-1];

int T = 1;

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

num[i] = i;

while(T--)

{

random\_shuffle(num,num+n);

printf("num: ");

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

printf("%d ",num[i]);

printf("\n");

get\_hash(Hash(num),ans);

printf("ans: ");

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

printf("%d ",ans[i]);

printf("\n\n\n");

}

return 0;

}

## 数位dp（考虑前导零）

int dfs(int len, int s, bool fp)

{

if (len == 0) return 1;

if (!fp && s >= 0 && dp[len][s] != -1) return dp[len][s];

int res = 0;

int fmax = fp? num[len] : 9;

for (int i = 0; i <= fmax; i++)

{

if (s == -1 && i == 0)

res += dfs(len-1, -1, fp && i == fmax);

else if (s != i)

res += dfs(len-1, i, fp&& i == fmax);

}

return fp || s < 0 ? res : dp[len][s] = res;

}

int solve(int n)

{

int len = 0;

while(n != 0)

{

num[++len] = n%10;

n /= 10;

}

return dfs(len, -1, 1);

}

## 四边形不等式

当函数w(i,j)满足 w(a,c)+w(b,d) <= w(b,c)+w(a,d) 且a<=b< c <=d 时，我们称w(i,j)满足四边形不等式。。

当函数w(i, j)满足w(i', j) <= w(i, j'); i <= i' < j <= j' 时，称w关于关于区间包含关系单 调。

s(i, j)=k是指m(i, j)这个状态的最优决策

以上定理的证明自己去查些资料

今天看得lrj的书中介绍的 四边形优化 做个笔记，加强理解

最有代价用d[i,j]表示

d[i,j]=min{d[i,k-1]+d[k+1,j]}+w[i,j]

其中w[i,j]=sum[i,j]

四边形不等式

w[a,c]+w[b,d]<=w[b,c]+w[a,d](a<b<c<d) 就称其满足凸四边形不等式

决策单调性

w[i,j]<=w[i',j'] ([i,j]属于[i',j']) 既 i'<=i<j<=j'

于是有以下三个定理

定理一： 如果w同时满足四边形不等式 和 决策单调性 ,则d也满足四边形不等式

定理二：当定理一的条件满足时，让d[i,j]取最小值的k为K[i,j]，则K[i,j-1]<=K[i,j]<=K[i+1,j]

定理三：w为凸当且仅当w[i,j]+w[i+1,j+1]<=w[i+1,j]+w[i,j+1]

由定理三知 判断w是否为凸即判断 w[i,j+1]-w[i,j]的值随着i的增加是否递减

于是求K值的时候K[i,j]只和K[i+1,j] 和 K[i,j-1]有关，所以 可以以i-j递增为顺序递推各个状态值最终求得结果 将O(n^3)转为O(n^2)

for(i = 0; i <= N; i ++)

{

f[i][i] = 0;

K[i][i] = i;

}

for(p = 1; p <= N - P; p ++)

{

for(i = 0; (j = i + p) <= N; i ++)

f[i][j] = INF;

for(i = 1; (j = i + p) <= N; i ++)

{

for(k = K[i][j - 1]; k <= K[i + 1][j]; k ++)

if((t = f[i - 1][k - 1] + getw(k, j)) < f[i][j])

{

f[i][j] = t;

K[i][j] = k;

}

}

}

## 斜率优化dp

#include <cstdio>

#include <cstring>

#include <iostream>

#include <algorithm>

#include <cmath>

#include <vector>

#include <queue>

#include <stack>

#include <set>

#include <map>

#include <tr1/unordered\_set>

#include <tr1/unordered\_map>

#include <bitset>

#include <cassert>

//#pragma comment(linker, "/STACK:1024000000,1024000000")

#define lson l, m, rt<<1

#define rson m+1, r, rt<<1|1

#define inf 1e9

#define debug(a) cout << #a" = " << (a) << endl;

#define debugarry(a, n) for (int i = 0; i < (n); i++) { cout << #a"[" << i << "] = " << (a)[i] << endl; }

#define clr(x, y) memset(x, y, sizeof x)

#define LL long long

#define uLL unsigned LL

using namespace std;

const int maxn = 500000+30;

LL X[maxn] , Y[maxn];

int he,ta;

void push(LL x,LL y)

{

while( he>ta+1 && (Y[he-1]-Y[he-2])\*(x-X[he-2]) >= (y-Y[he-2])\*(X[he-1]-X[he-2]) )

he--;

X[he] = x;

Y[he] = y;

he++;

}

LL cal(int i,LL k){

return Y[i] - k\*X[i];

}

LL Find(LL k)

{

while( he>ta+1 && cal(ta+1,k) <= cal(ta,k) )

ta++;

return cal(ta,k);

}

int main()

{

///freopen("input.txt", "r", stdin);

int n,m;

while(~scanf("%d%d",&n,&m))

{

int a;

LL sum=0;

he=ta=0;

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

{

scanf("%d",&a);

sum+=a;

LL t = sum\*sum+m;

if(he>ta)

t = min( t , Find(2\*sum) + sum\*sum + m );

push( sum , t + sum\*sum );

if( i == n-1 )

printf("%lld\n",t);

}

}

return 0;

}

/\*\*\*\*\*\*\*\*\*\*

公式为0

dp[i] = dp[j] + sum[i]\*sum[i] - 2\*sum[i]\*sum[j] + sum[j]\*sum[j] + m;

dp[j] + sum[j] \* sum[j] + m = 2 \* sum[i] \* sum[j] + dp[i] - sum[i] \* sum[i];

y = dp[j] + sum[j] \* sum[j] + m

k = 2 \* sum[i]

x = sum[j]

d = dp[i] - sum[i] \* sum[i]

\*\*\*\*\*\*\*\*\*\*\*/

## 舞蹈链

精确覆盖

const int maxn = 1000\*100\*2+30;

struct sad{

int L[maxn],R[maxn],U[maxn],D[maxn];

int Row[maxn],Col[maxn];

int h[maxn],s[maxn];

int si;

vector<int>ans;

void init(int m)

{

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

{

U[i] = D[i] = i;

L[i] = i-1;

R[i] = i+1;

s[i] = 0;

}

L[0] = m;

R[m] = 0;

clr(h,-1);

si = m;

ans.clear();

}

void link(int row,int col)

{

si++;

Col[si] = col; Row[si] = row;

s[col]++;

U[si] = U[col]; D[si] = col;

D[U[col]] = si;

U[col] = si;

if(h[row] != -1)

{

R[si] = h[row];

L[si] = L[h[row]];

R[L[si]] = si;

L[R[si]] = si;

}

else

h[row] = L[si] = R[si] = si;

}

void Remove(int c)

{

L[R[c]] = L[c];

R[L[c]] = R[c];

for(int i=D[c];i!=c;i=D[i]){

for(int j=R[i];j!=i;j=R[j]){

U[D[j]] = U[j];

D[U[j]] = D[j];

s[Col[j]]--;

}

}

}

void Resume(int c)

{

for(int i=U[c];i!=c;i=U[i]){

for(int j=L[i];j!=i;j=L[j]){

U[D[j]] = j;

D[U[j]] = j;

s[Col[j]]++;

}

}

R[L[c]] = c;

L[R[c]] = c;

}

bool dfs(){

if( R[0] == 0 )

return true;

int c = R[0];

// for(int i=R[0];i!=0;i=R[i])

// {

// if(!c||s[c]>s[i])

// c=i;

// }

Remove(c);

for(int i=D[c];i!=c;i=D[i])

{

ans.push\_back(Row[i]);

for(int j=R[i];j!=i;j=R[j])

Remove(Col[j]);

if(dfs()) return true;

for(int j=L[i];j!=i;j=L[j])

Resume(Col[j]);

ans.pop\_back();

}

Resume(c);

return false;

}

}dlx;

模糊覆盖

const int maxn = 15\*15\*15\*15\*2+100;

struct sad

{

int L[maxn],R[maxn],U[maxn],D[maxn];

int Row[maxn] , Col[maxn];

int h[maxn] , s[maxn];

int si,ans;

void init(int m){

for(int i=0;i<=m;i++){

U[i] = D[i] = i;

L[i] = i-1;

R[i] = i+1;

s[i] = 0;

}

L[0] = m;

R[m] = 0;

clr(h,-1);

si=m;

ans=inf;

}

void link(int row,int col){

si++;

Col[si] = col;

Row[si] = row;

s[col]++;

U[si] = U[col];

D[si] = col;

D[U[col]] = si;

U[col] = si;

if(h[row]!=-1)

{

R[si] = h[row];

L[si] = L[h[row]];

R[L[si]] = si;

L[R[si]] = si;

}

else

h[row] = L[si] = R[si] = si;

}

void Remove(int c)

{

for(int i=D[c];i-c;i=D[i]){

L[R[i]] = L[i];

R[L[i]] = R[i];

}

}

void Resume(int c)

{

for(int i=U[c];i!=c;i=U[i]){

L[R[i]] = R[L[i]] = i;

}

}

int vis[maxn];

int H()

{

int ret=0;

for(int i=R[0];i;i=R[i])

vis[i] = 0;

for(int i=R[0];i;i=R[i]) if(!vis[i]){

ret++;

vis[i] = 1;

for(int j=D[i];j!=i;j=D[j])

for(int k=R[j];k-j;k=R[k])

vis[Col[k]] = 1;

}

return ret;

}

void dfs(int cnt)

{

if(cnt+H()>=ans) return ;

if(R[0]==0){

ans = min(ans,cnt);

return ;

}

int c=R[0];

for(int i=R[0];i;i=R[i])

if(s[c]>s[i]) c=i;

for(int i=D[c];i-c;i=D[i]){

Remove(i);

for(int j=R[i];j-i;j=R[j])

Remove(j);

dfs(cnt+1);

for(int j=L[i];j-i;j=L[j])

Resume(j);

Resume(i);

}

}

}dlx;

## 左偏树

#include <iostream>

#include <vector>

using namespace std;

const int maxn = 2e5;

/// 0代表空 ， 每次弹出最大的元素

template<typename T>

struct lttree{

int lc[maxn] , rc[maxn] , dist[maxn],si;

T key[maxn];

void orderput(int rt){

if(rt){

orderput(lc[rt]);

printf("%d,",key[rt]);

orderput(rc[rt]);

}

}

int merge(int a,int b){

if(!a) return b;

if(!b) return a;

if( key[a] < key[b] ) swap(a,b);

rc[a] = merge(rc[a],b);

if( !lc[a] || dist[rc[a]] > dist[lc[a]] )

swap(lc[a],rc[a]);

dist[a] = rc[a]?dist[rc[a]]+1:0;

return a;

}

void push(int &rt,T x){

++si;

key[si] = x;

lc[si] = rc[si] = dist[si] = 0;

rt=merge(rt,si);

}

T &top(int &rt){

return key[rt];

}

void pop(int &rt){

rt=merge(lc[rt],rc[rt]);

}

};

lttree<pair<int,int> >LT;

char s[20] , q[20];

/\*\*\*

- push A x: insert x into stack A

- pop A: remove the top element of stack A

- merge A B: merge stack A and B

\*\*\*/

int main() {

// freopen( "in.txt", "r", stdin );

int n,x,a,b,CASE=0;

while(~scanf("%d",&n)){

if(!n) break;

a=b=0;LT.si=1;

printf("Case #%d:\n",++CASE);

for(int i=0;i<n;i++){

scanf("%s%s",s,q);

int &p = (\*q=='A'?a:b) , &ot = (\*q=='A'?b:a);

if(s[1]=='u'){

scanf("%d",&x);

LT.push(p,make\_pair(i,x));

}else if(s[1]=='o')

printf("%d\n",LT.key[p].second),LT.pop(p);

else{

scanf("%s",q);

p = LT.merge(p,ot);

ot = 0;

}

}

}

return 0;

}

## kdtree带修改

#include<cstdio>

#include<cstring>

#include<cstdlib>

#include<cmath>

#include<iostream>

#include<algorithm>

#include<queue>

#define inf (1e9+20)

#define ll long long

#define ls ch[x][0]

#define rs ch[x][1]

using namespace std;

const int maxn = 2e5;

///实验证明,a∈[0.55,0.75]是最快的.而且a越大,插入就越快,询问就越慢,反过来也一样.

struct P{

int a[2] , num;

ll dis;

}p[maxn];

int now , k = 2;

bool cmp(int a,int b){

return p[a].a[now] < p[b].a[now];

}

ll dist(P a,P b){

ll sum = 0;

for(int i=0;i<k;i++){

sum += 1ll\*(a.a[i]-b.a[i])\*(a.a[i]-b.a[i]);

}

return sum;

}

int ch[maxn][2] , co[maxn] , root;

void pushup(int x){

co[x] = co[ls] + co[rs] + 1;

}

const double alp = 0.75;

bool isbad(int x){

return max(co[ls],co[rs]) > alp\*co[x] + 5;

}

void build(int &x,int l,int r,int a[],int no){

if( l > r ){

x = 0;

return ;

}

int mid = l+r>>1;

now = no;

nth\_element(a+l,a+mid,a+r+1,cmp);

x = a[mid];

build(ls,l,mid-1,a,no^1);

build(rs,mid+1,r,a,no^1);

pushup(x);

}

P ans;

void Find(int x,int y,int no){

///如果想找出距离前k大的，请和陈队的板子配套使用

if(!x) return ;

ll dis = dist(p[y],p[x]);

/\*\*这一段即为找到点，可随意更改\*/

if( x != y ){

if( dis < ans.dis || (dis==ans.dis&&p[x].num < ans.num) ){

ans = p[x];

ans.dis = dis;

}

}

/\*\*\*/

ll d = p[y].a[no] - p[x].a[no];

Find(ch[x][d>0],y,no^1);

if( d\*d <= ans.dis )

Find(ch[x][d<=0],y,no^1);

}

/\*\*\*/

int Fa , No;

int Insert(int &x,int fa,int k,int no){

if( x == 0 ){

x = k;

return 0;

}

int t = p[x].a[no]<p[k].a[no];

int r = Insert(ch[x][t],x,k,no^1);

pushup(x);

if( isbad(x) ) r = x , Fa = fa , No = no;

return r;

}

int su[maxn] , len;

void travel(int x){

if(x==0) return ;

travel(ls);

su[len++] = x;

travel(rs);

}

void Insert(int k){

int x = Insert(root,0,k,0);

if(x){

int tmp = x;

len = 0;

travel(x);

build(x,0,len-1,su,No);

if(Fa) ch[Fa][ch[Fa][1]==tmp] = x;

else root = x;

}

}

/\*\*\*/

int si;

void newnode(int x,int y,int id){

si++;

p[si].a[0] = x;

p[si].a[1] = y;

p[si].num = id;

ch[si][0] = ch[si][1] = 0;

co[si] = 1;

}

void init(){

si = 0;

ch[0][0] = ch[0][1] = 0;

co[0] = 0;

root = 0;

}

int num[maxn];

int main()

{

freopen("in.txt","r",stdin);

int T,n;

scanf("%d",&T);

while(T--){

scanf("%d",&n);

init();

int x,y;

for(int i=1;i<=n;i++){

scanf("%d%d",&x,&y);

newnode(x,y,i);

Insert(si);

}

for(int i=1;i<=n;i++){

ans.dis = 4e18;

Find(root,i,0);

printf("%lld\n",ans.dis);

}

}

return 0;

}

# 树

## 树分治

const int maxn = 1e5;

struct edge{

int to,next;

}G[maxn<<2];

int h[maxn],si;

void add(int u,int v)

{

G[si].to=v;

G[si].next=h[u];

h[u]=si++;

}

int done[maxn];

int sum[maxn];

int fN;

int dfs1(int u,int pa)

{

sum[u]=1;

for(int i=h[u];~i;i=G[i].next){

int v = G[i].to;

if( done[v] || pa == v ) continue;

sum[u] += dfs1( v , u );

}

return sum[u];

}

int dfs2(int u,int pa)

{

for(int i=h[u];~i;i=G[i].next){

int v = G[i].to;

if( done[v] || pa == v ) continue;

if( sum[v] \* 2 >= fN ) return dfs2(v,u);

}

return u;

}

void dfs(int u)

{

fN = dfs1(u,-1);

int r = dfs2(u,-1);

sol(r);

done[r]=1;

for(int i=h[r];~i;i=G[i].next)

{

int v = G[i].to;

if( done[v] == 0 )

dfs(v);

}

}

void solve()

{

clr(done,0);

dfs(1);

}

## 树链剖分

点查询

const int maxn = 50000+200;

struct sad

{

int to,next;

}G[maxn<<2];

int h[maxn],si;

void add(int u,int v)

{

G[si].to=v;

G[si].next=h[u];

h[u]=si++;

}

int siz[maxn],dep[maxn];

int fa[maxn],son[maxn],top[maxn];

void dfs1(int u,int f,int d)

{

fa[u] = f;

dep[u] = d;

siz[u] = 1;

son[u] = -1;

for(int i=h[u];~i;i=G[i].next){

int v=G[i].to;

if( v != f )

{

dfs1(v,u,d+1);

siz[u] += siz[v];

if(son[u]==-1||siz[son[u]]<siz[v])

son[u]=v;

}

}

}

int p[maxn],fp[maxn],pos;

void dfs2(int u,int sf)

{

top[u]=sf;

p[u]=pos++;

fp[p[u]]=u;

if( son[u] == -1 ) return ;

dfs2(son[u],sf);

for(int i=h[u];~i;i=G[i].next)

{

int v=G[i].to;

if(son[u]!=v&&fa[u]!=v)

dfs2(v,v);

}

}

int rs[maxn];

void upp(int x,int p)

{

while(x<=pos)

{

rs[x]+=p;

x+=x&-x;

}

}

int get(int x)

{

int ret=0;

while(x)

{

ret += rs[x];

x-=x&-x;

}

return ret;

}

void Change(int u,int v,int d)

{

int f1=top[u] , f2=top[v];

while(f1 != f2){

if( dep[f1] < dep[f2] )

{

swap(f1,f2);

swap(u,v);

}

upp(p[f1],d);

upp(p[u]+1,-d);

u=fa[f1];

f1=top[u];

}

if( dep[u] > dep[v] ) swap(u,v);

upp(p[u],d);

upp(p[v]+1,-d);

}

void init()

{

clr(h,-1);

si=0;

pos=1;

}

---------------------------------------------------------------------------------------------------------------------------

边操作

struct sad{

int to,next,id;

}G[maxn<<2];

int h[maxn],si;

void add(int u,int v,int id)

{

G[si].to=v;

G[si].id=id;

G[si].next=h[u];

h[u]=si++;

}

int siz[maxn],dep[maxn];

int fa[maxn],son[maxn],top[maxn];

int fid[maxn];

void dfs1(int u,int f,int d)

{

// debug(u);

fa[u]=f;

dep[u]=d;

siz[u]=1;

son[u]=-1;

for(int i=h[u];~i;i=G[i].next){

int v=G[i].to;

// debug(v);

if(v!=f){

fid[G[i].id]=v;

dfs1(v,u,d+1);

siz[u] += siz[v];

if(son[u]==-1||siz[son[u]]<siz[v])

son[u]=v;

}

}

}

int p[maxn],fp[maxn],pos;

void dfs2(int u,int sf)

{

top[u]=sf;

p[u]=pos++;

fp[p[u]]=u;

if(son[u]==-1) return ;

dfs2(son[u],sf);

for(int i=h[u];~i;i=G[i].next)

{

int v=G[i].to;

if(son[u]!=v&&fa[u]!=v)

dfs2(v,v);

}

}

int rs[maxn];

void upp(int x,int p)

{

while(x<=pos)

{

rs[x]+=p;

x+=x&-x;

}

}

int get(int x)

{

int ret=0;

while(x)

{

ret += rs[x];

x-=x&-x;

}

return ret;

}

int Query(int u,int v)

{

int ret=0,f1=top[u],f2=top[v];

while(f1!=f2)

{

if(dep[f1]<dep[f2])

{

swap(f1,f2);

swap(u,v);

}

ret += get(p[u]) - get(p[f1]-1);

u=fa[f1];

f1=top[u];

}

if(u==v) return ret;

if(dep[u]>dep[v]) swap(u,v);

ret += get(p[v]) - get(p[son[u]]-1);

return ret;

}

void init()

{

clr(h,-1);

si=0;

pos=1;

}

---------------------------------------------------------------------------------------------------------------------------

带顺序的点操作

const int maxn = 100000+20;

struct edge{

int to,next;

}G[maxn<<2];

int h[maxn],si;

void add(int u,int v){

G[si].to=v;

G[si].next=h[u];

h[u]=si++;

}

int siz[maxn],dep[maxn];

int fa[maxn],son[maxn],top[maxn];

void dfs1(int u,int f,int d){

fa[u]=f;

dep[u]=d;

siz[u]=1;

son[u]=-1;

for(int i=h[u];~i;i=G[i].next){

int v=G[i].to;

if(v^f){

dfs1(v,u,d+1);

siz[u]+=siz[v];

if(son[u]==-1||siz[son[u]]<siz[v])

son[u]=v;

}

}

}

int p[maxn],fp[maxn],pos;

void dfs2(int u,int sf){

top[u]=sf;

p[u]=pos++;

fp[p[u]]=u;

if(son[u]==-1) return ;

dfs2(son[u],sf);

for(int i=h[u];~i;i=G[i].next){

int v=G[i].to;

if(son[u]!=v&&fa[u]!=v)

dfs2(v,v);

}

}

struct sad{

int s,lc,rc;

sad(){s=-1;}

sad(int c)

{

s=1;

lc=rc=c;

}

sad operator+(sad b)

{

if( s == -1 ) return b;

if( b.s == -1 ) return \*this;

sad c;

c.s = s + b.s + ((rc==b.lc)?-1:0);

c.lc = lc;

c.rc = b.rc;

return c;

}

sad rev()

{

sad c=\*this;

swap(c.lc,c.rc);

return c;

}

void print()

{

printf("s %d lc %d rc %d\n",s,lc,rc);

}

}rs[maxn<<2];

int col[maxn<<2];

void pushup(int rt)

{

rs[rt] = rs[rt<<1] + rs[rt<<1|1];

}

void pushdown(int rt)

{

if(col[rt]!=-1){

rs[rt<<1] = sad(col[rt]);

rs[rt<<1|1] = sad(col[rt]);

col[rt<<1] = col[rt<<1|1] = col[rt];

col[rt] = -1;

}

}

void build(int l,int r,int rt){

col[rt]=-1;

if(l==r)

{

rs[rt]=sad(0);

return ;

}

int m=l+r>>1;

build(lson);

build(rson);

pushup(rt);

}

void update(int L,int R,int c,int l,int r,int rt)

{

if( L <= l && r <= R )

{

rs[rt] = sad(col[rt]=c);

return ;

}

pushdown(rt);

int m=l+r>>1;

if( L <= m ) update(L,R,c,lson);

if( m < R ) update(L,R,c,rson);

pushup(rt);

}

sad query(int L,int R,int l,int r,int rt){

if( L <= l && r <= R )

return rs[rt];

pushdown(rt);

int m=l+r>>1;

sad ret;

if( L <= m ) ret = ret + query(L,R,lson);

if( m < R ) ret = ret + query(L,R,rson);

return ret;

}

void Change(int u,int v,int c){

int f1=top[u],f2=top[v];

while(f1!=f2){

if(dep[f1]<dep[f2]){

swap(f1,f2);

swap(u,v);

}

update(p[f1],p[u],c,1,pos-1,1);

u=fa[f1];

f1=top[u];

}

if(dep[u]>dep[v]) swap(u,v);

update(p[u],p[v],c,1,pos-1,1);

}

int Query(int u,int v)

{

int f1=top[u],f2=top[v];

sad ans1,ans2;

while(f1!=f2){

if(dep[f1]<dep[f2]){

swap(f1,f2);

swap(u,v);

swap(ans1,ans2);

}

ans1 = query(p[f1],p[u],1,pos-1,1) + ans1;

u=fa[f1];

f1=top[u];

}

if(dep[u]>dep[v])

{

swap(u,v);

swap(ans1,ans2);

}

ans2 = query(p[u],p[v],1,pos-1,1) + ans2;

return (ans1.rev()+ans2).s;

}

void init()

{

clr(h,-1);

si=0;

pos=1;

}

int w[maxn];

int main()

{

//freopen("input.txt","r",stdin);

int n,m;

while(~scanf("%d%d",&n,&m)){

init();

for(int i=1;i<=n;i++)

scanf("%d",&w[i]);

int a,b;

char c;

for(int i=0;i<n-1;i++){

scanf("%d%d",&a,&b);

add(a,b);

add(b,a);

}

dfs1(1,-1,1);

dfs2(1,1);

build(1,pos-1,1);

/\*for(int i=1;i<=n;i++)

printf("%d top %d\n",i,top[i]);\*/

for(int i=1;i<=n;i++)

update(p[i],p[i],w[i],1,pos-1,1);

for(int i=0;i<m;i++){

scanf(" %c",&c);

if( c == 'Q' )

{

scanf("%d%d",&a,&b);

printf("%d\n",Query(a,b));

}else{

int v;

scanf("%d%d%d",&a,&b,&v);

Change(a,b,v);

}

}

}

return 0;

}

## 树上莫队（附树上启发式合并）

#include<cstdio>

#include<cmath>

#include<cstring>

#include<cstdlib>

#include<iostream>

#include<algorithm>

#include<vector>

#include<queue>

#include<map>

#include<complex>

#include<list>

#include<set>

#include<assert.h>

#include<bitset>

#define lson l,mid,ls[rt]

#define rson mid+1,r,rs[rt]

using namespace std;

typedef long long ll;

using namespace std;

inline bool read(int &n){

int x=0,tmp=1;

char c=getchar();

while( (c<'0'||c>'9') && c!='-'&&c!=EOF ) c=getchar();

if(c==EOF) return false;

if(c=='-') c=getchar(),tmp=-1;

while(c>='0'&&c<='9') x\*=10,x+=(c-'0'),c=getchar();

n=x\*tmp;

return true;

}

inline void write(ll n){

if(n<0){

putchar('-');

n=-n;

}

int len=0,date[20];

while(n){

date[len++] = n%10;

n/=10;

}

if(!len) date[len++] = 0;

while(len--) putchar(date[len]+'0');

}

const int maxn = 100000 + 200;

struct edge{

int to,next;

}G[maxn<<1];

int h[maxn],si;

void add(int u,int v){

G[si].to=v;

G[si].next=h[u];

h[u]=si++;

}

int val[maxn] , n , sub[maxn] , len;

ll ans[maxn];

vector<pair<int,int> >q1[maxn];

vector<ll>tmp[maxn];

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*树上启发式合并\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*begin

struct \_\_sad{

int h[maxn];

ll cnt1[maxn];

int rc[maxn\*23] , ls[maxn\*23] , rs[maxn\*23];

int sz;

void pushup(int rt){

}

void pushdown(int rt){

}

void init(){

rc[0] = ls[0] = rs[0] = 0;

sz = 1;

for(int i=0;i<=n;i++) cnt1[i] = 0;

}

void update(int p,int x,int l,int r,int &rt){

rc[sz]=rc[rt];

ls[sz]=ls[rt];

rs[sz]=rs[rt];

rt=sz++;

if(l==r){

cnt1[rc[rt]]-=sub[l];

cnt1[rc[rt]+=x]+=sub[l];

return ;

}

int mid=l+r>>1;

pushdown(rt);

if(p<=mid) update(p,x,lson);

else update(p,x,rson);

pushup(rt);

}

void megre(int l,int r,int &rt,int rt2){

if(rt==0) rt=rt2;

else if(rt2==0) ;

else{

if(l==r){

cnt1[rc[rt]]-=sub[l];

cnt1[rc[rt2]]-=sub[l];

cnt1[rc[rt]+=rc[rt2]]+=sub[l];

return ;

}

int mid=l+r>>1;

pushdown(rt);

pushdown(rt2);

megre(lson,ls[rt2]);

megre(rson,rs[rt2]);

pushup(rt);

}

}

}bit;

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*树上启发式合并\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*end

void dfs1(int u,int fa){

// printf("u %d fa %d\n",u,fa);

bit.h[u] = 0;

tmp[u].clear();

for(int i=0;i<q1[u].size();i++){

int a = q1[u][i].first , id = q1[u][i].second;

tmp[u].push\_back( bit.cnt1[a] );

// printf("<%lld>\n",bit.cnt1[a]);

}

for(int i=h[u];~i;i=G[i].next){

int v = G[i].to;

if(v!=fa){

dfs1(v,u);

bit.megre(0,n,bit.h[u],bit.h[v]);

}

}

bit.update(val[u],1,0,n,bit.h[u]);

for(int i=0;i<q1[u].size();i++){

int a = q1[u][i].first , id = q1[u][i].second;

ans[id] = \_\_gcd(ans[id],bit.cnt1[a]-tmp[u][i]);

// printf("(%d)\n",bit.cnt1[a]);

}

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*树上莫队\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*begin

int block\_size , be[maxn] , siz[maxn] , sblo;

int pos[maxn] , dep[maxn] , fa[maxn][23] , cur\_time;

bool in[maxn];

struct sad{

int u,v,a,id;

sad(){};

sad(int u,int v,int a,int id):u(u),v(v),a(a),id(id){

if(pos[u]>pos[v]) swap(u,v);

};

bool operator<(const sad &a)const{

if(be[u]==be[a.u]) return pos[v] < pos[a.v];

return be[u]<be[a.u];

}

};

vector<sad>q2;

void dfs2(int u,int f){

fa[u][0] = f;

pos[u] = ++cur\_time;

dep[u] = dep[f]+1;

for(int i=h[u];~i;i=G[i].next){

int v = G[i].to;

if(v==f) continue;

if( siz[be[u]] < block\_size )

++siz[be[u]] , be[v] = be[u];

else ++siz[++sblo] , be[v] = sblo;

dfs2(v,u);

}

}

int LCA(int u,int v){

if(dep[u]<dep[v]) swap(u,v);

for(int i=22;i>=0;i--)

if( dep[fa[u][i]] >= dep[v])

u = fa[u][i];

if(u==v) return u;

for(int i=22;i>=0;i--)

if( fa[u][i] != fa[v][i] )

u=fa[u][i] , v = fa[v][i];

return fa[u][0];

}

ll cnt2[maxn];

int rc[maxn];

void change(int u,int c){ ///c=1为加入，0为删除

cnt2[rc[val[u]]] -= sub[val[u]];

if(c) rc[val[u]]++;

else rc[val[u]]--;

cnt2[rc[val[u]]] += sub[val[u]];

// printf("u %d c %d rc %d %d\n",u,c,rc[val[u]],val[u]);

}

void initlca(){

for(int j=1;j<23;j++)

for(int i=1;i<=n;i++)

fa[i][j] = fa[fa[i][j-1]][j-1];

}

void work(int u,int v){

int anc = LCA(u,v);

for(;u!=anc;u=fa[u][0]){

in[u] ^= 1;

change(u,in[u]);

}

for(;v!=anc;v=fa[v][0]){

in[v] ^= 1;

change(v,in[v]);

}

}

void solve2(){

block\_size = sqrt(n+1)+1;

// cout<<"block::"<<block\_size<<endl;

siz[1] = 1;

be[1] = 1;

sblo = 1;

dep[0] = 0; fa[0][0] = 0;

dfs2(1,0);

initlca();

sort(q2.begin(),q2.end());

memset(in,0,sizeof in);

memset(cnt2,0,sizeof cnt2);

memset(rc,0,sizeof rc);

int l=1,r=1,anc;

for(int i=0;i<q2.size();i++){

// printf("u %d v %d\n",q2[i].u,q2[i].v);

work(l,q2[i].u);

work(r,q2[i].v);

l = q2[i].u , r = q2[i].v;

anc = LCA(l,r);

change(anc,1);

ans[q2[i].id] = \_\_gcd(ans[q2[i].id] , cnt2[q2[i].a] );

// cout<<" ss "<<cnt2[q2[i].a]<<' '<<q2[i].id<<endl;

// cout<<" ds"<<q2[i].a<<endl;

change(anc,0);

}

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*树上莫队\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*end

int main()

{

// freopen("in.txt","r",stdin);

int T,m,CASE=0;

read(T);

while(T--){

read(n);

read(m);

si=0;

memset(h,-1,sizeof h);

int op,u,v,a,b;

len = 0;

for(int i=1;i<=n;i++)

read(val[i]),sub[len++]=val[i];

sort(sub,sub+len);

len=unique(sub,sub+len) - sub;

for(int i=1;i<=n;i++)

val[i] = lower\_bound(sub,sub+len,val[i])-sub;

for(int i=0;i<n-1;i++){

read(u);

read(v);

add(u,v);

add(v,u);

}

for(int i=1;i<=n;i++) q1[i].clear();

q2.clear();

for(int i=0;i<m;i++) ans[i] = 0;

for(int i=0;i<m;i++){

read(op);

read(u);

read(v);

read(a);

read(b);

if(op==1){

q1[u].push\_back( make\_pair( a , i ) );

q1[u].push\_back( make\_pair( b , i ) );

}else{

q2.push\_back(sad(u,v,a,i));

q2.push\_back(sad(u,v,b,i));

}

}

bit.init();

dfs1(1,-1);

solve2();

printf("Case #%d:\n",++CASE);

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

write(ans[i]),putchar('\n');

}

return 0;

}

#include <stdio.h>

#include <string.h>

#include <iostream>

#include <vector>

#include <algorithm>

#include <bitset>

#include <set>

#include <assert.h>

#define lson l,m,rt<<1

#define rson m+1,r,rt<<1|1

using namespace std;

#define ll long long

const int maxn = 1e5+1e4;

vector<int>G[maxn];

int block\_size , be[maxn] , siz[maxn] , sblo;

int pos[maxn] , dep[maxn] , fa[maxn][23] , cur\_time;

bool in[maxn];

struct sad{

int u,v,ti;

bool operator<(const sad &A)const{

if(be[u]!=be[A.u]) return be[u] < be[A.u];

if(be[v]!=be[A.v]) return be[v] < be[A.v];

return ti < A.ti;

}

};

vector<sad>q;

void dfs(int u,int f){

fa[u][0] = f;

for(int i=1;i<23;i++)

fa[u][i] = fa[fa[u][i-1]][i-1];

pos[u] = ++cur\_time;

dep[u] = dep[f]+1;

for(int i=0;i<G[u].size();i++){

int v = G[u][i];

if(v==f) continue;

if( siz[be[u]] < block\_size )

++siz[be[u]] , be[v] = be[u];

else

++siz[++sblo] , be[v] = sblo;

dfs(v,u);

}

}

int LCA(int u,int v){

if(dep[u]<dep[v]) swap(u,v);

for(int i=22;i>=0;i--)

if( dep[fa[u][i]] >= dep[v] )

u = fa[u][i];

if(u==v) return u;

for(int i=22;i>=0;i--)

if( fa[u][i] != fa[v][i] )

u=fa[u][i],v=fa[v][i];

return fa[u][0];

}

int ficol[maxn];

int col[maxn] , cnt[maxn];

vector<int>pp;

void change(int u,int c){

// printf("u %d in %d\n",u,c);

if(c==1) cnt[col[u]]++;

else cnt[col[u]]--;

if(cnt[col[u]]&1)

pp.push\_back(col[u]);

}

void work(int u,int v){

int anc = LCA(u,v);

for(;u!=anc;u=fa[u][0])

change(u,in[u]^=1);

for(;v!=anc;v=fa[v][0])

change(v,in[v]^=1);

}

int x[maxn] , y[maxn] , ans[maxn];

int n,m;

void solve(){

block\_size = pow(n,2.0/3)+2;

siz[1] = be[1] = sblo = 1;

dep[0] = fa[0][0] = 0;

dfs(1,0);

sort(q.begin(),q.end());

for(int i=0,j;i<q.size();i=j){

for(j=i;j<q.size() && q[i].ti == q[j].ti;j++)

;

memcpy(col,ficol,sizeof col);

memset(in,0,sizeof in);

memset(cnt,0,sizeof cnt);

int l=1,r=1,anc,ti = 0;

pp.clear();

for(int k=i;k<j;k++){

// printf("ti %d %d\n",ti,q[k].ti);

while( ti < q[k].ti ){

if(x[ti]!=-1){

if(in[x[ti]]) change(x[ti],0);

col[x[ti]] = y[ti];

if(in[x[ti]]) change(x[ti],1);

}

ti++;

}

work(l,q[k].u);

work(r,q[k].v);

l = q[k].u;

r = q[k].v;

anc = LCA(l,r);

change(anc,1);

// printf("l %d r %d\n",l,r);

while(pp.size() && cnt[pp.back()]%2==0) pp.pop\_back();

if(pp.size() == 0)

y[q[k].ti] = -1;

else

y[q[k].ti] = pp.back();

change(anc,0);

}

}

}

int main()

{

// freopen("avl.out","w",stdout);

// for(int i=0;i<20;i++)

// printf("%d %d %d\n",rand()%2,rand()%10+1,rand()%10+1);

// return 0;

// printf("%d\n",(15\*(sizeof st))>>10);

// freopen("in.txt","r",stdin);

int T;

scanf("%d",&T);

while(T--){

scanf("%d%d",&n,&m);

for(int i=1;i<=n;i++)

G[i].clear();

int a,b,c;

for(int i=0;i<n-1;i++){

scanf("%d%d",&a,&b);

G[a].push\_back(b);

G[b].push\_back(a);

}

for(int i=1;i<=n;i++)

scanf("%d",&ficol[i]);

q.clear();

for(int i=0;i<m;i++){

scanf("%d%d%d",&a,&x[i],&y[i]);

if(a==1){

q.push\_back(sad());

q.back().u = x[i];

q.back().v = y[i];

q.back().ti = i;

x[i] = -1;

}

}

solve();

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

if(x[i]==-1)

printf("%d\n",y[i]);

}

return 0;

}

## 基环外向树

/\*\*

一种解法，把剪掉一条边，算

另一种解法，作成树形dp，再考虑环

\*\*/

/\*\*

找一个最短路径走过所有点

1

10

7 1

5 3

4 5

8 9

6 4

8 6

2 7

10 3

4 1

9 7

\*\*/

#include<cstdio>

#include<cmath>

#include<cstring>

#include<cstdlib>

#include<iostream>

#include<iomanip>

#include<algorithm>

#include<vector>

#include<queue>

#include<map>

#include<complex>

#include<set>

#include<assert.h>

#include<bitset>

#define lson l,m,rt<<1

#define rson m+1,r,rt<<1|1

#define ls ch[x][0]

#define rs ch[x][1]

using namespace std;

#define inf (1e9+1e8)

const int maxn = 3e5;

bool vs[maxn];

vector<int>p,q,G[maxn];

bool dfscir(int u,int fa){

p.push\_back(u);

vs[u] = true;

for(int i=0;i<G[u].size();i++){

int v = G[u][i];

if(v!=fa){

if(vs[v]){

while(p.back()!=v)

q.push\_back(p.back()) , p.pop\_back();

q.push\_back(v);

return true;

}

if( dfscir(v,u) ) return true;

}

}

p.pop\_back();

return false;

}

pair<int,pair<int,int> >ans;

int dp1[maxn] , dp2[maxn] , dp3[maxn];

int ro1[maxn];

pair<int,int>ro3[maxn];

void dfs(int u,int fa,int fa2){

dp2[u] = 0;

vector<pair<int,int> >p1;

vector<pair<int,pair<int,int> > >p3;

for(int i=0;i<G[u].size();i++){

int v = G[u][i];

if( v != fa && v != fa2 ){

dfs(v,u,-1);

dp2[u] += dp2[v] + 2;

p1.push\_back(make\_pair(dp1[v]-dp2[v],ro1[v]));

p3.push\_back(make\_pair(dp3[v]-dp2[v],ro3[v]));

}

}

sort(p1.begin(),p1.end());

sort(p3.begin(),p3.end());

dp1[u] = dp3[u] = dp2[u];

ro1[u] = u , ro3[u] = make\_pair(u,u);

if( p1.size() ){

ro1[u] = p1[0].second;

dp1[u] += p1[0].first - 1;

}

if( p1.size() >= 2 ){

pair<int,int>ta = make\_pair(p1[0].second,p1[1].second);

if(ta.first > ta.second)

swap(ta.first,ta.second);

dp3[u] += p1[0].first + p1[1].first - 2;

ro3[u] = ta;

}

if(p3.size()){

int t = dp2[u] + p3[0].first;

if( dp3[u] > t ){

dp3[u] = t;

ro3[u] = p3[0].second;

}else if( dp3[u] == t )

ro3[u] = min(ro3[u],p3[0].second);

}

}

int main()

{

// freopen("in.txt","r",stdin);

int T,n,CASE=0;

scanf("%d",&T);

while(T--){

scanf("%d",&n);

while(n<=2);

int a,b;

for(int i=1;i<=n;i++)

G[i].clear();

p.clear(),q.clear();

memset(vs,0,sizeof vs);

for(int i=1;i<=n;i++){

scanf("%d%d",&a,&b);

G[a].push\_back(b);

G[b].push\_back(a);

}

dfscir(1,-1);

int sz = q.size() , sum = (sz-1)\*2;

vector<pair<int,pair<int,int> > >p3;

for(int i=0;i<sz;i++){

int u = q[i];

dfs(q[i],q[(i+1)%sz],q[(i+sz-1)%sz]);

sum += dp2[u];

p3.push\_back( make\_pair( dp3[u] - dp2[u] , ro3[u] ) );

q.push\_back(u);

}

sort(p3.begin(),p3.end());

ans = make\_pair( sum + p3[0].first - sz + 2, p3[0].second );

priority\_queue<pair<pair<int,int>,int>,vector<pair<pair<int,int>,int> >,greater<pair<pair<int,int>,int> > >Q;

for(int i=0;i<q.size();i++){

int u = q[i];

while( Q.size() && Q.top().second <= i - sz )

Q.pop();

if(Q.size()){

int s = sum + Q.top().first.first - i + dp1[u] - dp2[u];

pair<int,int>ta = make\_pair(ro1[u],Q.top().first.second);

if(ta.first > ta.second)

swap(ta.first,ta.second);

ans = min(ans,make\_pair(s,ta));

}

Q.push(make\_pair(make\_pair(dp1[u] - dp2[u] + i , ro1[u]) , i) );

}

printf("Case #%d: %d %d %d\n",++CASE,ans.first,ans.second.first,ans.second.second);

}

return 0;

}

## 虚树

3 3

Chen

Qian

Zhuge

1 2

2 3

Chen Chen

Chen Sun

Zhuge Chen

4 2

Chen

Chen

Qian

Qian

1 2

2 3

1 4

Chen Qian

Qian Qian

///询问两种颜色的点的最远距离

#include <cstdio>

#include <iostream>

#include <algorithm>

#include <set>

#include <stack>

#include <map>

#include <vector>

#include <deque>

#include <string.h>

#define lson l,m,rt<<1

#define rson m+1,r,rt<<1|1

#define inf 1e9

using namespace std;

#define ll long long

const int maxn = 1e5+300;

vector<int>G[maxn];

int n , col[maxn] , dep[maxn] ,a,b;

int fa[maxn][26];

int in[maxn] , curtime;

void dfs(int u,int pa,int d){

dep[u] = d;

fa[u][0] = pa;

in[u] = ++curtime;

for(int i=1;i<26;i++)

fa[u][i] = fa[fa[u][i-1]][i-1];

for(int i=0;i<G[u].size();i++)

if( G[u][i] != pa )

dfs(G[u][i],u,d+1);

}

int lca(int u,int v){

if( dep[u] < dep[v] ) swap(u,v);

for(int i=25;i>=0;i--)

if( dep[u] >= dep[v] + (1<<i) )

u = fa[u][i];

for(int i=25;i>=0;i--)

if( fa[u][i] != fa[v][i] )

u = fa[u][i] , v = fa[v][i];

return u==v?u:fa[u][0];

}

struct sad{

vector<pair<int,int> >G[maxn];

int ans;

int la[maxn] , lb[maxn];

void init(){

ans = -inf;

}

void add(int u,int v){

// if( u == v ) return ;

G[u].push\_back( make\_pair( v, dep[v] - dep[u] ) );

printf("u %d v %d\n",u,v);

}

void dfs(int u){

la[u] = lb[u] = -inf;

if( col[u] == a ) la[u] = 0;

if( col[u] == b ) lb[u] = 0;

ans = max(ans,la[u]+lb[u]+1);

for(int i=0;i<G[u].size();i++){

int v = G[u][i].first , w = G[u][i].second;

dfs(v);

ans = max(ans,la[u]+lb[v]+w+1);

ans = max(ans,lb[u]+la[v]+w+1);

la[u] = max(la[v]+w,la[u]);

lb[u] = max(lb[v]+w,lb[u]);

}

G[u].clear();

}

}SA;

char s[100];

map<string,int>name;

map<ll,int>Ans;

vector<int>pp[maxn];

int si;

bool cmp(const int &a,const int &b){

return in[a] < in[b];

}

int main()

{

freopen("in.txt","r",stdin);

int n,q;

while(~scanf("%d%d",&n,&q)){

name.clear();

Ans.clear();

si = 0;

for(int i=1;i<=n;i++){

scanf("%s",s);

if( !name.count(s) ){

name[s] = ++si;

pp[si].clear();

}

col[i] = name[s];

pp[name[s]].push\_back(i);

}

for(int i=0;i<n-1;i++){

scanf("%d%d",&a,&b);

G[a].push\_back(b);

G[b].push\_back(a);

}

curtime = 0;

dfs(1,-1,0);

for(int i=0;i<q;i++){

scanf("%s",s);

a = name.count(s)?name[s]:-1;

scanf("%s",s);

b = name.count(s)?name[s]:-1;

if( a == -1 || b == -1 ){

printf("-1\n");

continue;

}

if( a > b ) swap(a,b);

ll has = 1ll\*a\*n+b;

if( Ans.count(has) == 0 ){

SA.init();

vector<int>p = pp[a];

for(int i=0;i<pp[b].size();i++)

p.push\_back(pp[b][i]);

cout<<p.size()<<endl;

/\*\*\*\*/

sort(p.begin(),p.end(),cmp);

for(int i=0,sz=p.size();i<sz-1;i++)

p.push\_back( lca(p[i],p[i+1]) );

sort(p.begin(),p.end(),cmp);

stack<int>Q;

for(int i=0;i<p.size();i++){

int t = p[i];

if( !i || t != p[i-1] ){

while(Q.size()&&Q.top()!=lca(Q.top(),t))

Q.pop();

if( Q.size() ){

SA.add(Q.top(),t);

}

Q.push(t);

}

}

SA.dfs(p[0]);///p[0] is root

/\*\*\*\*\*\*\*/

Ans[has] = SA.ans;

}

printf("%d\n",Ans[has]);

}

}

return 0;

}

# 数论

## 快速幂与和

struct mat

{

int m[100][100];

};

int n;

mat mul(mat a,mat b)

{

mat c;

clr(c.m,0);

for(int k=1;k<=n;k++)

for(int i=1;i<=n;i++) if(a.m[i][k])

for(int j=1;j<=n;j++)

c.m[i][j] += a.m[i][k] \* b.m[k][j];

}

mat add(mat a,mat b)

{

mat c;

for(int i=1;i<=n;i++)

for(int j=1;j<=n;j++)

c.m[i][j] = a.m[i][j] + b.m[i][j];

}

mat re;

mat sum\_pow\_mat(mat a,int n)

{

if(n%2)

{

if(n==1) return re=a;

mat p = sum\_pow\_mat(a,n-1);

re = mul(re,a);

return add(a,mul(a,p));

}

else

{

mat p = sum\_pow\_mat(a,n/2);

mat ans = add(p,mul(p,re));

re = mul(re,re);

return ans;

}

}

## Fft

const double PI = acos(-1.0);

const int maxn = 5e5+30;//开四倍

///typedef complex<double> Comp;

///结构体会慢一倍左右

struct Comp{

double r,i;

Comp(double \_r=0,double \_i=0){r=\_r,i=\_i;}

Comp operator+(const Comp &b)const{

return Comp(r+b.r,i+b.i);

}

Comp operator-(const Comp &b)const{

return Comp(r-b.r,i-b.i);

}

Comp operator\*(const Comp &b)const{

return Comp(r\*b.r-i\*b.i,r\*b.i+i\*b.r);

}

};

void Change(Comp \*y,int len){

for(int i=1,j=len/2;i<len-1;i++){

if(i<j) swap(y[i],y[j]);

int k = len/2;

while(j>=k){

j -= k;

k /= 2;

}

if(j<k) j+=k;

}

}

void fft(Comp \*a,int len,int on){

Change(a,len);

for(int h=1;h<len;h<<=1){

Comp wn(cos(-on\*PI/h),sin(-on\*PI/h));

Comp w(1,0);

for(int k=0;k<h;k++){

for(int j=k;j<len;j+=h\*2){

Comp u = a[j];

Comp v = w\*a[j|h];

a[j] = u + v;

a[j|h] = u - v;

}

w = w\*wn;

}

}

if(on==-1)

for(int i=0;i<len;i++)

a[i].r /= len;

}

ll pow\_mod(ll a,ll n,ll mod){

ll r = 1;

while(n){

if(n&1) r=r\*a%mod;

a=a\*a%mod;

n>>=1;

}

return r;

}

Comp b1[2][maxn] , b2[2][maxn] , q[maxn];

int m,p;

void FFT2(ll \*a1,ll \*a2,ll \*a,int len1,int len2){

int len = 1;

while(len<2\*len1||len<2\*len2) len<<=1;

int M = sqrt(mod+10) + 10;

for(int i=0;i<len;i++)

b1[0][i] = i<len1?a1[i] % M : 0,

b1[1][i] = i<len1?a1[i] / M : 0;

for(int i=0;i<len;i++)

b2[0][i] = i<len2?a2[i] % M :0,

b2[1][i] = i<len2?a2[i] / M : 0;

fft(b1[0],len,1);

fft(b1[1],len,1);

fft(b2[0],len,1);

fft(b2[1],len,1);

for(int x=0;x<2;x++)

for(int y=0;y<2;y++){

for(int i=0;i<len;i++)

q[i] = b1[x][i] \* b2[y][i];

fft(q,len,-1);

for(int i=0;i<len;i++)

a[i] = ( a[i] + (ll)(q[i].r+0.5)%mod\*pow\_mod(M,x+y,mod)%mod )%mod;

}

}

ll inv(ll a,ll mod){

return pow\_mod(a,mod-2,mod);

}

int a[maxn] , b[maxn];

ll tmp[maxn] , tmp2[maxn];

ll q1[maxn];

void mul(int \*a,int \*b,int \*c,int k)

{

int t = pow\_mod(p,k,m);

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

tmp[i] = a[i];

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

tmp2[1ll\*i\*t%m] = b[i];

for(int i=0;i<2\*m;i++) q1[i] = 0;

FFT2(tmp,tmp2,q1,m,m);

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

c[i] = (q1[i] + q1[i+m])%mod;

}

int solve(int n)

{

memset(a,0,sizeof a);

memset(b,0,sizeof b);

b[0] = 1;

for(int i='A'; i<='Z'; i++)

a[i%m]++;

int k = 1;

while(n)

{

if(n&1)

mul(a,b,b,k);

mul(a,a,a,k);

n>>=1;

k<<=1;

}

int ans = 0;

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

ans = ( ans + 1ll\*b[i]\*(b[i]-1+mod)%mod \* inv(2,mod)%mod )%mod;

return ans;

}

int main()

{

// freopen("in.txt","r",stdin);

int n;

while(~scanf("%d%d%d",&n,&m,&p))

{

printf("%d\n",solve(n));

}

return 0;

}

## Fwt

const int maxn = 1<<21;

const int mod = 1e9+7;

int pow\_mod(int a,int n){

int r = 1;

while(n){

if(n&1) r=(ll)r\*a%mod;

a = (ll)a\*a%mod;

n>>=1;

}

return r;

}

int inv2 = pow\_mod(2,mod-2);

void fwt(int \*a,int l,int r){

if( l == r ) return ;

int mid=l+r>>1;

fwt(a,l,mid); fwt(a,mid+1,r);

for(int i=l;i<=mid;i++){

int u = a[i] , v = a[i+mid-l+1];

a[i] = (u+v)%mod;

a[i+mid-l+1] = (u-v+mod)%mod;

}

}

void ifwt(int \*a,int l,int r){

if( l == r ) return ;

int mid=l+r>>1;

for(int i=l;i<=mid;i++){

int u = a[i] , v = a[i+mid-l+1];

a[i] = 1ll\*(u+v)%mod\*inv2%mod;

a[i+mid-l+1] = 1ll\*(u-v+mod)%mod\*inv2%mod;

}

ifwt(a,l,mid); ifwt(a,mid+1,r);

}

int N = 1<<len;

void FWT(int \*a,int \*b){

fwt(a,0,N-1);

fwt(b,0,N-1);

for(int i=0;i<N;i++)

a[i] = 1ll \* a[i] \* b[i] % mod;

ifwt(a,0,N-1);

}

## Ntt

#include<cstdio>

#include<cmath>

#include<cstring>

#include<cstdlib>

#include<iostream>

#include<algorithm>

#include<vector>

#include<queue>

#include<map>

#include<complex>

#include<set>

#include<assert.h>

#include<bitset>

#define lson l,m,rt<<1

#define rson m+1,r,rt<<1|1

#define inf 1e9

using namespace std;

#define ll long long

const ll mod = 50000000001507329LL; // 5e16 , G = 3

//const LL mod = 4179340454199820289LL; // 29 \* (2 ^ 57), 4e18, g = 3

//const LL mod = 1945555039024054273LL; // 27 \* (2 ^ 56), 1e18, g = 5

//const int mod = 4179340454199820289; // g = 3

//const int mod = 1945555039024054273; // g = 3

int MOD = 1e6+3;

const int maxn=30000\*5;

int G = 3;

ll mul\_mod( ll x, ll y, ll p )

{///mul\_mod只能在p<=5e16时使用

ll tmp=(x\*y-((ll)((double)x\*y/p+1e-6 )\*p))%p;

return tmp<0?tmp+p:tmp;

}

/\*

ll mul\_mod(ll a,ll b,ll mod)

{

ll r = 0;

while(b)

{

if(b&1){

r += a;

if( r > mod ) r -= mod;

}

a<<=1;

if( a > mod ) a -= mod;

b>>=1;

}

return r;

}

\*/

ll pow\_mod(ll a,ll n,ll mod){

ll r = 1;

while(n){

if(n&1) r=mul\_mod(r,a,mod);

a=mul\_mod(a,a,mod);

n>>=1;

}

return r;

}

ll inv(ll a,ll mod){

return pow\_mod(a,mod-2,mod);

}

#define LL long long

LL mul( LL x, LL y, LL p )

{

LL tmp=(x\*y-((LL)((double)x\*y/p+1e-6 )\*p))%p;

return tmp<0?tmp+p:tmp;

}

void Change(ll y[],int len)

{

for(int i=1,j=len/2; i<len-1; i++)

{

if(i<j) swap(y[i],y[j]);

int k=len/2;

while(j>=k)

{

j-=k;

k/=2;

}

if(j<k) j+=k;

}

}

inline void ntt(LL \*a, int n, int flag)

{

int aaa = 0;

Change(a,n);

for (int k=1;k<n;k<<=1)

{

LL wn=pow\_mod(G, (mod-1)/(k<<1), mod), w=1;

if (flag==-1) wn=pow\_mod(wn, mod-2, mod);

for (int i=0;i<k;i++,w=mul(w,wn,mod))

for (int j=i;j<n;j+=(k<<1))

{

aaa++;

LL x=a[j], y=mul(w,a[j|k],mod);

a[j]=x+y;

while (a[j]>=mod) a[j]-=mod;

a[j|k]=x-y+mod;

while (a[j|k]>=mod) a[j|k]-=mod;

}

}

if (flag==-1)

{

LL inv=pow\_mod(n, mod-2, mod);

for (int i=0;i<n;i++) a[i]=mul(a[i],inv,mod);

}

}

void NTT(ll \*a1,ll \*a2,ll \*a,

int len1,int len2,int len = 0)

{

len = 1;

while( len < 2\*len1 || len < 2\*len2 ) len<<=1;

for(int i=len1; i<len; i++) a1[i] = 0;

for(int i=len2; i<len; i++) a2[i] = 0;

ntt(a1,len,1);

ntt(a2,len,1);

for(int i=0; i<len; i++)

a1[i] = mul\_mod(a1[i],a2[i],mod);

ntt(a1,len,-1);

for(int i=0; i<len; i++)

a[i] = a1[i]%mod;

}

int a[maxn] , b[maxn];

ll tmp[maxn] , tmp2[maxn];

ll q1[maxn];

int m,p;

void mul(int \*a,int \*b,int \*c,int k)

{

int t = pow\_mod(p,k,m);

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

tmp[i] = a[i];

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

tmp2[1ll\*i\*t%m] = b[i];

NTT(tmp,tmp2,q1,m,m);

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

c[i] = (q1[i] + q1[i+m])%mod%MOD;

}

int solve(int n)

{

memset(a,0,sizeof a);

memset(b,0,sizeof b);

b[0] = 1;

for(int i='A'; i<='Z'; i++)

a[i%m]++;

int k = 1;

while(n)

{

if(n&1)

mul(a,b,b,k);

mul(a,a,a,k);

n>>=1;

k<<=1;

}

int ans = 0;

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

ans = ( ans + 1ll\*b[i]\*(b[i]-1+MOD)%MOD \* inv(2,MOD)%MOD )%MOD;

return ans;

}

int main()

{

// freopen("in.txt","r",stdin);

int n;

while(~scanf("%d%d%d",&n,&m,&p))

{

printf("%d\n",solve(n));

}

return 0;

}

## Lucas

const int maxn = 110119 +100;

const int mod = 110119;

int A[maxn];

int pow\_mod(int a,int n){

int r = 1;

while(n){

if(n&1) r=(ll)r\*a%mod;

a=(ll)a\*a%mod;

n>>=1;

}

return r;

}

int inv(int a){return pow\_mod(a,mod-2);}

void init(){

A[0] = 1;

for(int i=1;i<maxn;i++)

A[i] = (ll)A[i-1] \* i %mod;

}

int lucas(ll a,ll b){

int re = 1;

while(a&&b){

ll aa = a%mod , bb = b%mod;

if( aa < bb ) return 0;

re = (ll)re\*A[aa]%mod\*inv((ll)A[bb]\*A[aa-bb]%mod)%mod;

a/=mod;

b/=mod;

}

return re;

}

## lucas推广（数论小板子）

///扩展lucas时间复杂度较高，其实是O（mod）的。

const int maxn = 2e5;

int Len = 0;

int p[maxn] , xc[maxn] , xb[maxn] , cir[maxn] = {1,1};

void div\_fac(int m){

Len = 0;

for(int i=2;i\*i<=m;i++) if(m%i==0){

p[Len] = i;

xc[Len] = 1;

while(m%i==0) xc[Len] \*= i , m /= i;

Len++;

}

if(m>1) p[Len] = xc[Len++] = m;

}

int getcount(int x,int p){

int ret = 0;

while(x){

x /= p;

ret += x;

}

return ret;

}

int pow\_mod(int a,int n,int mod){

a%=mod;

int r = 1;

while(n){

if(n&1) r=(ll)r\*a%mod;

a=(ll)a\*a%mod;

n>>=1;

}

return r;

}

int gcd(int a,int b,int &x,int &y){

if(b==0){x=1;y=0;return a;}

int d = gcd(b,a%b,y,x);

y-=a/b\*x;

return d;

}

int inv(int a,int mod){

int x,y;

while( gcd(a,mod,x,y) != 1 );

return (x+mod)%mod;

}

int crt(int a[],int m[],int n){

int M = 1;

for(int i=0;i<n;i++) M\*=m[i];

int ret = 0;

for(int i=0;i<n;i++){

int x,y,tm=M/m[i];

gcd(tm,m[i],x,y);

ret=(ret+(ll)tm\*x%M\*a[i])%M;

}

return (ret+M)%M;

}

int lucas(int n,int m,int mod){

div\_fac(mod);

for(int i=0;i<Len;i++){

for(int j=2;j<xc[i];j++)

if(j%p[i]==0) cir[j] = cir[j-1];

else cir[j] = (ll)cir[j-1]\*j%xc[i];

xb[i] = 1;

int tmp = n;

while(tmp){

xb[i] = (ll)xb[i]\*pow\_mod(cir[xc[i]-1],tmp/xc[i],xc[i])%xc[i]

\*cir[tmp-tmp/xc[i]\*xc[i]]%xc[i];

tmp /= p[i];

}

for(int j=2;j<xc[i];j++)

if(j%p[i]==0) cir[j] = cir[j-1];

else cir[j] = (ll)cir[j-1]\*inv(j,xc[i])%xc[i];

tmp = n-m;

while(tmp){

xb[i] = (ll)xb[i]\*pow\_mod(cir[xc[i]-1],tmp/xc[i],xc[i])%xc[i]

\*cir[tmp-tmp/xc[i]\*xc[i]]%xc[i];

tmp /= p[i];

}

tmp = m;

while(tmp){

xb[i] = (ll)xb[i]\*pow\_mod(cir[xc[i]-1],tmp/xc[i],xc[i])%xc[i]

\*cir[tmp-tmp/xc[i]\*xc[i]]%xc[i];

tmp /= p[i];

}

int fct = getcount(n,p[i])-getcount(m,p[i])-

getcount(n-m,p[i]);

while(fct--) xb[i] = (xb[i]\*p[i])%xc[i];

}

return crt(xb,xc,Len);

}

## n以内素数

#include <cstdio>

#include <iostream>

#include <algorithm>

#include <set>

#include <map>

#include <vector>

#include <deque>

#include <string.h>

#define lson l,m,rt<<1

#define rson m+1,r,rt<<1|1

#define inf (1e18)

using namespace std;

#define ll long long

///数组大小开成sqrt(n) , 时间复杂度为n^(3/4)， 但还是比较快的。

const int maxn = sqrt(1e12);

ll f[maxn] , g[maxn];

ll solve(ll n){

ll m;

ll sum = 0;

for(m=1;m\*m<=n;m++)

f[m] = n/m-1,sum++;

for(ll i=1;i<=m;i++) g[i] = i-1,sum++;

for(ll i=2;i<=m;i++){

if(g[i]==g[i-1]) continue;

for(ll j=1;j<=m-1&&j<=n/i/i;j++){

if(i\*j<m) f[j] -= f[i\*j] - g[i-1];

else f[j] -= g[n/i/j] - g[i-1];

sum++;

}

for(ll j=m;j>=i\*i;j--)

g[j] -= g[j/i] - g[i-1],sum++;

}

printf("sum %lld\n",sum);

return f[1];

}

int main()

{

// freopen("in.txt","r",stdin);

ll n;

while(cin>>n)

cout<<solve(n)<<endl;

return 0;

}

#include <iostream>

#include <algorithm>

#include <set>

#include <map>

#include <vector>

#include <deque>

#include <string.h>

#define lson l,m,rt<<1

#define rson m+1,r,rt<<1|1

#define inf (1e18)

using namespace std;

#define ll long long

///数组大小开成sqrt(n) , 时间复杂度为n^(2/3)， 但还是比较快的。

const int N = sqrt(1e12);

bool vis[N];

int Len,pri[N] , pi[N];

void init(){

Len = 0;

memset(vis,0,sizeof vis);

vis[0] = vis[1] = true;

for(int i=1;i<N;i++){

if(!vis[i]){

pri[++Len] = i;

for(ll j=1ll\*i\*i;j<N;j+=i) vis[j] = true;

}

pi[i] = Len;

}

}

const int M = 7;

const int PM = 2\*3\*5\*7\*11\*13\*17;

int phi[PM+1][M+1] , sz[M+1];

void init2(){

init();

sz[0] = 1;

for(int i=0;i<=PM;i++) phi[i][0] = i;

for(int i=1;i<=M;i++){

sz[i] = pri[i] \* sz[i-1];

for(int j=1;j<=PM;++j)

phi[j][i] = phi[j][i-1] - phi[j/pri[i]][i-1];

}

}

int sqrt2(ll x){

ll r = sqrt(x-0.1);

while(r\*r<=x) r++;

return r-1;

}

int sqrt3(ll x){

ll r = cbrt(x-0.1);

while(r\*r\*r<=x) r++;

return r-1;

}

ll getphi(ll x,int s){

if(s==0) return x;

if(s<=M) return phi[x%sz[s]][s] + x/sz[s]\*phi[sz[s]][s];

if(x<=pri[s]\*pri[s]) return pi[x] - s + 1;

if(x<pri[s]\*pri[s]\*pri[s]&&x<N){

ll s2x = pi[sqrt2(x)];

ll ans = pi[x] - (s2x+s-2)\*(s2x-s+1)/2;

for(int i=s+1;i<=s2x;i++) ans+=pi[x/pri[i]];

return ans;

}

return getphi(x,s-1) - getphi(x/pri[s],s-1);

}

ll getpi(ll x){

if(x<N) return pi[x];

ll ans = getphi(x,pi[sqrt3(x)]) + pi[sqrt3(x)] - 1;

for(int i=pi[sqrt3(x)]+1,ed=pi[sqrt2(x)];i<=ed;i++)

ans -= getpi(x/pri[i]) - i + 1;

}

ll lehmer\_pi(ll x){

if(x<N) return pi[x];

int a = lehmer\_pi(sqrt2(sqrt2(x)));

int b = lehmer\_pi(sqrt2(x));

int c = lehmer\_pi(sqrt3(x));

ll sum = getphi(x,a) + (ll)(b+a-2)\*(b-a+1)/2;

for(int i=a+1;i<=b;i++){

ll w = x/pri[i];

sum -= lehmer\_pi(w);

if(i>c) continue;

ll lim = lehmer\_pi(sqrt2(w));

for(int j=i;j<=lim;j++)

sum -= lehmer\_pi(w/pri[j]) -j+1;

}

return sum;

}

int main()

{

// freopen("in.txt","r",stdin);

init2();

ll n;

while(cin>>n)

cout<<lehmer\_pi(n)<<endl;

return 0;

}

## Schreier-Sims

#pragma comment(linker, "/STACK:1024000000,1024000000")

#include<cstdio>

#include<cmath>

#include<cstring>

#include<cstdlib>

#include<iostream>

#include<algorithm>

#include<vector>

#include<queue>

#include<map>

#include<stack>

#include<complex>

#include<list>

#include<set>

#include<assert.h>

#include<bitset>

#define lson l,m,rt<<1

#define rson m+1,r,rt<<1|1

using namespace std;

typedef long long ll;

struct Per

{

vector<int> P;

Per() {} Per(int n)

{

P.resize(n);

}

Per inv()const

{

Per ret(P.size());

for(int i = 0; i < int(P.size()); ++i) ret.P[P[i]] = i;

return ret;

}

int &operator [](const int &dn)

{

return P[dn];

}

void resize(const size\_t &sz)

{

P.resize(sz);

}

size\_t size()const

{

return P.size();

}

const int &operator [](const int &dn)const

{

return P[dn];

}

};

Per operator \*(const Per &a, const Per &b)

{

Per ret(a.size());

for(int i = 0; i < (int)a.size(); ++i) ret[i] = b[a[i]];

return ret;

}

typedef vector<Per> Buc;

typedef vector<int> Tab;

typedef pair<int,int> pii;

int n,m;

vector<Buc>buckets , bucketsinv;

vector<Tab> lookuptable;

int fastFilter(const Per &g, bool addToGroup = true)

{

int n = buckets.size();

Per p = g;

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

{

int res = lookuptable[i][p[i]];

if(res == -1)

{

if(addToGroup)

{

buckets[i].push\_back(p);

bucketsinv[i].push\_back(p.inv());

lookuptable[i][p[i]] = (int)buckets[i].size() - 1;

}

return i;

}

p = p \* bucketsinv[i][res];

}

return -1;

}

long long calcTotalSize()

{

long long ret = 1;

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

ret \*= buckets[i].size();

// cout<<"\*"<<buckets[i].size()<<endl;

return ret;

}

bool inGroup(const Per &g)

{

return fastFilter(g, false) == -1;

}

void solve(const Buc &gen,int \_n) // m perm[0..n - 1]s

{

n = \_n, m = gen.size();

{

//clear all

vector<Buc> \_buckets(n);

swap(buckets, \_buckets);

vector<Buc> \_bucketsInv(n);

swap(bucketsinv, \_bucketsInv);

vector<Tab> \_lookupTable(n);

swap(lookuptable, \_lookupTable);

}

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

{

lookuptable[i].resize(n);

fill(lookuptable[i].begin(), lookuptable[i].end(), -1);

}

Per id(n);

for(int i = 0; i < n; ++i) id[i] = i;

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

{

buckets[i].push\_back(id);

bucketsinv[i].push\_back(id);

lookuptable[i][i] = 0;

}

for(int i = 0; i < m; ++i) fastFilter(gen[i]);

queue<pair<pii,pii> > toUpdate;

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

for(int j = i; j < n; ++j)

for(int k = 0; k < (int)buckets[i].size(); ++k)

for(int l = 0; l < (int)buckets[j].size(); ++l)

toUpdate.push(make\_pair(pii(i,k), pii(j,l)));

while(!toUpdate.empty())

{

pii a = toUpdate.front().first, b = toUpdate.front().second;

toUpdate.pop();

int res=fastFilter(buckets[a.first][a.second]\*buckets[b.first][b.second]);

if(res==-1) continue;

pii newPair(res, (int)buckets[res].size() - 1);

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

for(int j = 0; j < (int)buckets[i].size(); ++j)

{

if(i <= res) toUpdate.push(make\_pair(pii(i, j), newPair));

if(res <= i) toUpdate.push(make\_pair(newPair, pii(i, j)));

}

}

}

int main()

{

// freopen("in.txt","r",stdin);

int r,n;

scanf("%d%d",&r,&n);

Buc gen;

Per p(n);

while(r--)

{

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

scanf("%d",&p[i]),p[i]--;

gen.push\_back(p);

}

solve(gen,n);

printf("%lld\n",calcTotalSize());

return 0;

}

## 斯特林数

///第一类斯特林数

///s(n,m) = s(n-1,m-1) + (n-1)\*s(n-1,m)

///第二类斯特林数

///S(n,m) = S(n-1,m-1) + m\*S(n-1,m)

///第二类斯特林数

LL S(int n,int m)

{

LL ret = 0;

for(int k=0;k<=m;k++)

{

LL t = C(m,k) \* pow\_mod(m-k,n) %mod;

if( k % 2 == 0 )

ret = (ret + t) %mod;

else

ret = ((ret-t)%mod+mod)%mod;

}

ret = ret \* rA[m] %mod;

return ret;

}

## 五边形数

#include <iostream>

#include <string.h>

#include <stdio.h>

using namespace std;

typedef long long LL;

const int N=100005;

const LL MOD=1000000007;

LL ans[N],tmp[N];

void Init()

{

int t=1000;

for(int i=-1000;i<=1000;i++)

tmp[i+t]=i\*(3\*i-1)/2;

ans[0]=1;

for(int i=1;i<N;i++)

{

ans[i]=0;

for(int j=1;j<=i;j++)

{

if(tmp[j+t]<=i)

{

if(j&1) ans[i]+=ans[i-tmp[j+t]];

else ans[i]-=ans[i-tmp[j+t]];

}

else break;

ans[i]=(ans[i]%MOD+MOD)%MOD;

if(tmp[t-j]<=i)

{

if(j&1) ans[i]+=ans[i-tmp[t-j]];

else ans[i]-=ans[i-tmp[t-j]];

}

else break;

}

ans[i]=(ans[i]%MOD+MOD)%MOD;

}

}

int main()

{

int t,n;

Init();

cin>>t;

while(t--)

{

cin>>n;

cout<<ans[n]<<endl;

}

return 0;

}

问一个数n能被拆分成多少种方法，且每一种方法里数字重复个数不能超过k（等于k）

#include <iostream>

#include <string.h>

#include <stdio.h>

using namespace std;

const int N = 100005;

const int MOD = 1000000007;

int dp[N];

void Init()

{

dp[0] = 1;

for(int i=1;i<N;i++)

{

dp[i] = 0;

for(int j=1;;j++)

{

int t = (3\*j-1)\*j / 2;

if(t > i) break;

int tt = dp[i-t];

if(t+j <= i) tt = (tt + dp[i-t-j])%MOD;

if(j&1) dp[i] = (dp[i] + tt)%MOD;

else dp[i] = (dp[i] - tt + MOD)%MOD;

}

}

}

int Work(int n,int k)

{

int ans = dp[n];

for(int i=1;;i++)

{

int t = k\*i\*(3\*i-1) / 2;

if(t > n) break;

int tt = dp[n-t];

if(t + i\*k <= n) tt = (tt + dp[n-t-i\*k])%MOD;

if(i&1) ans = (ans - tt + MOD)%MOD;

else ans = (ans + tt)%MOD;

}

return ans;

}

int main()

{

Init();

int n,k,t;

scanf("%d",&t);

while(t--)

{

scanf("%d%d",&n,&k);

printf("%d\n",Work(n,k));

}

return 0;

}

## 线性基

关于线性基的学习与理解

1、线性基：

　　若干数的线性基是一组数a1,a2,...an，其中ax的最高位的1在第x位。通过线性基中元素xor出的数的值域与原来的数xor出数的值域相同。

2、线性基的构造法：

　　对每一个数p从高位到低位扫，扫到第x位为1时，若ax不存在，则ax=p并结束此数的扫描，否则令p=p xor ax。

3、查询：

　　用线性基求这组数xor出的最大值：从高往低扫ax，若异或上ax使答案变大，则异或。

4、判断：

　　用线性基求一个数能否被xor出：从高到低，对该数每个是1的位置x，将这个数异或上ax（注意异或后这个数为1的位置和原数就不一样了），若最终变为0，则可被异或出。需要特判0（在构造过程中看是否有p变为0即可）。例子：(11111,10001)的线性基是a5=11111，a4=01110，要判断11111能否被xor出，11111 xor a5=0，则这个数后来就没有是1的位置了，最终得到结果为0，说明11111能被xor出。

个人谈一谈对线性基的理解:

　　很多情况下，只有有关异或运算和求最值，就可以用到线性 基。线性基有很多很好的性质，比如说如果有很多个数，我们可以构出这些数的线性基，那么这个线性基可以通过互相xor，能够构出原来的数可以相互xor构出的所有的数。所以可以大大减少判断的时间和次数。同时线性基的任何一个非空子集都不会使得其xor和为0，证明也很简单，反证法就可以说明。这个性质在很多题目中可以保证算法合法性，比如：BZOJ2460。

　　构造的方法有点像贪心，从大到小保证高位更大。也比较好理解。就是这几行代码：

复制代码

1 for(int i=1;i<=n;i++) {

2

3 　　　　for(int j=62;j>=0;j--) {

4

5 　　　　 if(!(a[i]>>j)) continue;//对线性基的这一位没有贡献

6

7 　　　　　 　if(!p[j]) { p[j]=a[i]; break; }//选入线性基中

8

9 　　　　　　 a[i]^=p[j];

10

11 　　　　 }

12

13 　　　}

复制代码

　　可以把n个数变成只有二进制位数那么多的数，这就是线性基的优秀之处。

　　查询的话，也是一个贪心思想，如果可以使得ans更大，就把这一位的基xor进ans。

　　for(int i=62;i>=0;i--) if((ans^p[i])>ans) ans=ans^p[i];//从线性基中得到最大值

　　这就是线性基的基本用法和个人的一些理解。

## 圆的反演

【HDU】4773 Problem of Apollonius#include<cstdio>

#include<stdio.h>

#include<cmath>

#include<cstring>

#include<cstdlib>

#include<iostream>

#include<algorithm>

#include<vector>

#include<queue>

#include<map>

#include<complex>

#include<set>

#include<assert.h>

#include<bitset>

#define lson l,m,rt<<1

#define rson m+1,r,rt<<1|1

using namespace std;

#define ll long long

#define inf 1e9

double const eps = 1e-8;

typedef complex<double> point;

double det(point a,point b){ return (conj(a)\*b).imag();}

//叉积，顺时针为负数，逆时针为正数|a|×|b|×sin(ang)

double dot(point a,point b){ return (a\*conj(b)).real();}

//点积|a|×|b|×cos(ang)

inline int sgn(double n){ return abs(n)<eps?0:(n<0?-1:1);}

point trans(point a){return point(-a.imag(),a.real());}

struct circle{

point c;

double r;

circle(){}

circle(point c,double r):c(c),r(r){}

};

struct line: public vector<point>

{

line(){}

line(point a,point b){ push\_back(a),push\_back(b);}

};

point vec(point a){ return a/abs(a);}

point vec(const line &a){ return a[1]-a[0];}

point tc;//反演中心，半径为1

circle inverse(circle u)

{

circle T;

double t = abs(u.c-tc);

double x = 1.0 / (t - u.r);

double y = 1.0 / (t + u.r);

T.r = (x - y) / 2.0;

double s = (x + y) / 2.0;

T.c = tc + (u.c - tc) \* (s / t);

return T;

}

circle inverse(const line &u)

{

point p=u[1]-u[0];

double t = det(tc-u[0],p);

if(t < 0) t = -t;

double d = abs(p);

t /= d;

if(t > eps)

{

double w = 0.5 / t;

point dir = trans(p);

point a1 = tc + dir \* (w / d);

point b1 = tc - dir \* (w / d);

if(sgn(abs(det(p,a1-u[0]))-abs(det(p,b1-u[0])) )<0 )

return circle(a1,w);

else return circle(b1,w);

}

return circle(point(0,0),-1);

}

line inverseL(circle u)

{

point c=2.0\*(u.c-tc);

point p=trans(c-tc);

c/=norm(c);

return line(tc+c,tc+c+p);

}

void getOutCL(circle u, circle v,line &ret1,line &ret2)//外公切线

{

double l=abs(u.c-v.c);

double d=abs(u.r-v.r);

//cout<<d<<endl;

// cout<<l<<"\*"<<endl;

double ang=acos(d/l);

if(sgn(u.r-v.r)>0)swap(u,v);

point p=u.c-v.c;

p=vec(p)\*u.r;

ret1=line(u.c+rotate(p,ang),u.c+rotate(p,-ang));

p=vec(p)\*v.r;

ret2=line(v.c+rotate(p,ang),v.c+rotate(p,-ang));

swap(ret1[1],ret2[0]);

}

void getInCL(circle u, circle v,line &ret1,line &ret2)//内公切线

{

double l=abs(u.c-v.c);

double d=l\*u.r/(u.r+v.r);

double tmp=u.r/d;

//tmp=fix(tmp);

double ang=acos(tmp);

point p=v.c-u.c;

p=vec(p)\*u.r;

ret1=line(u.c+rotate(p,ang),u.c+rotate(p,-ang));

p=u.c-v.c;

p=vec(p)\*v.r;

ret2=line(v.c+rotate(p,ang),v.c+rotate(p,-ang));

swap(ret1[1],ret2[0]);

}

## 单纯型法

　　申奥成功后，布布经过不懈努力，终于成为奥组委下属公司人力资源部门的主管。布布刚上任就遇到了一个难

题：为即将启动的奥运新项目招募一批短期志愿者。经过估算，这个项目需要N 天才能完成，其中第i 天至少需要

Ai 个人。 布布通过了解得知，一共有M 类志愿者可以招募。其中第i 类可以从第Si 天工作到第Ti 天，招募费用

是每人Ci 元。新官上任三把火，为了出色地完成自己的工作，布布希望用尽量少的费用招募足够的志愿者，但这

并不是他的特长！于是布布找到了你，希望你帮他设计一种最优的招募方案。

Input

　　第一行包含两个整数N, M，表示完成项目的天数和可以招募的志愿者的种类。 接下来的一行中包含N 个非负

整数，表示每天至少需要的志愿者人数。 接下来的M 行中每行包含三个整数Si, Ti, Ci，含义如上文所述。为了

方便起见，我们可以认为每类志愿者的数量都是无限多的。

Output

　　仅包含一个整数，表示你所设计的最优方案的总费用。

Sample Input

3 3

2 3 4

1 2 2

2 3 5

3 3 2

Sample Output

14

HINT

1 ≤ N ≤ 1000，1 ≤ M ≤ 10000，题目中其他所涉及的数据均 不超过2^31-1。

#include<cstdio>

#include<iostream>

#include<cstring>

using namespace std;

double A[10031][1031];

#define abs(a) (a<0?-a:a)

const

double eps=1e-5;

char c;

inline void read(int &a)

{

a=0;do c=getchar();while(c<'0'||c>'9');

while(c<='9'&&c>='0')a=(a<<3)+(a<<1)+c-'0',c=getchar();

}

int n,m;

inline void change(int x,int y)

{

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

if((i^x)&&abs(A[i][y])>eps)

{

double tp=A[i][y]/A[x][y];

for(int j=0;j<=n;j++)

A[i][j]-=tp\*A[x][j];

A[i][y]-=tp\*A[x][y];

}

}

const

int INF=1e9;

inline double simplex()

{

while(true)

{

int x=0;

for(int i=1;i<=n;i++)

if(A[0][i]>eps&&(x==0||A[0][i]>A[0][x])){x=i;}

if(x==0)return -A[0][0];

double Min=INF;

int tp=0;

for(int i=1;i<=m;i++)

if(A[i][x]>eps&&Min>A[i][0]/A[i][x])Min=A[i][0]/A[i][x],tp=i;

if(tp==0){A[0][x]=0;return -A[0][0];}

change(tp,x);

}

}

/// 求 max{cx | Ax ≤ b, x ≥ 0} 的解

/// 求 min{by | A'y >= c, y ≥ 0} 的解 二者是等价的

int main()

{

freopen("std.in","r",stdin);

freopen("self.out","w",stdout);

read(n),read(m);

int tp;

for(int i=1;i<=n;i++)

read(tp),A[0][i]=tp;

for(int i=1;i<=m;i++)

{

int k,s,t,c;

read(s),read(t);for(int j=s;j<=t;j++)A[i][j]=1;

read(c);A[i][0]=c;

}

double a=simplex();

printf("%d\n",(int)(a+eps));

return 0;

}

## 欧拉函数前缀和

#include<cstdio>

#include<cstring>

#include<algorithm>

#include<iostream>

#define fo(i,a,b) for(int i=a;i<=b;i++)

#define rep(i,a) for(int i=last[a];i;i=net[i])

#include <tr1/unordered\_map>

#define N 5000000

using namespace std;

typedef long long ll;

const int Mo=1000000007;

const int mo=2333333;

const int ni=500000004;

int phi[N+5],p[N+5];

bool bz[N+5];

tr1::unordered\_map<ll,int>Q;

ll calc(ll x) {

if (x<=N) return phi[x];int k=x%mo;ll ans=0,z=x%Mo;

if( Q.count(x) ) return Q[x];

for(ll l=2,r;l<=x;l=r+1) r=x/(x/l),(ans+=(r-l+1)%Mo\*calc(x/l)%Mo)%=Mo;

ans=(z\*(z+1)%Mo\*ni%Mo-ans+Mo)%Mo;

return Q[x] = ans;

}

/// ol(n) = n - sim{n|m}(ol(m))

/// mo(n) = [n==1] - sim{n|m}(mo(m))

/// time is O(n/sqrt(N),N) , when N = n^2/3 , time is n^2/3

int main() {

fo(i,2,N) {

if (!bz[i]) p[++p[0]]=i,phi[i]=i-1;

fo(j,1,p[0]) {

int k=i\*p[j];if (k>N) break;

bz[k]=1;if (!(i%p[j])) {phi[k]=phi[i]\*p[j];break;}

phi[k]=phi[i]\*(p[j]-1);

}

}

phi[1]=1;fo(i,1,N) (phi[i]+=phi[i-1])%=Mo;

ll n;

scanf("%lld",&n);

printf("%lld",calc(n));

}

## 矩阵的逆

import java.io.\*;

import java.util.\*;

import java.math.\*;

import java.text.\*;

public class Main {

public static int n;

public static BigInteger [][]a = new BigInteger[200][200];

public static BigInteger [][]A = new BigInteger[200][200];

public static BigInteger [][]B = new BigInteger[200][200];

public static BigInteger []C = new BigInteger[200];

public static BigInteger yy;

static Scanner cin = new Scanner(new BufferedInputStream(System.in));

static PrintWriter cout = new PrintWriter(new BufferedOutputStream(System.out));

public static void Gauss(){

int i,j,k,max\_r;

BigInteger tmp = new BigInteger("0");

BigInteger rans = new BigInteger("1");

BigInteger ans = new BigInteger("1");

for(k=0;k<n;k++){

max\_r = k;

for(i=k+1;i<n;i++)

if( a[i][k].abs().compareTo(a[max\_r][k].abs()) >= 0 )

max\_r = i;

if( max\_r != k ){

for(j=k;j<2\*n;j++){

tmp = a[k][j] ;

a[k][j] = a[max\_r][j] ;

a[max\_r][j] = tmp;

}

ans = ans.multiply(new BigInteger("-1"));

}

if( a[k][k].compareTo(new BigInteger("0")) == 0 ){

///无逆矩阵

continue;

}

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

if( i!=k && a[i][k].compareTo(new BigInteger("0")) != 0 ){

tmp = a[i][k].abs().gcd(a[k][k].abs());

BigInteger ta = a[k][k].abs().divide(tmp);

BigInteger tb = a[i][k].abs().divide(tmp);

if( a[i][k].multiply(a[k][k]).compareTo(new BigInteger("0")) < 0 )

tb = tb.multiply(new BigInteger("-1"));

for(j=0;j<2\*n;j++)

a[i][j] = a[i][j].multiply(ta).subtract(a[k][j].multiply(tb));

rans = rans.multiply(ta);

}

}

}

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

ans = ans.multiply(a[i][i]);

ans = ans.divide(rans);

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

for(j=0;j<n;j++)

a[i][j+n] = a[i][j+n].multiply(ans).divide(a[i][i]);

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

C[i] = BigInteger.valueOf(0);

for(int j=0;j<n;j++){

B[i][j] = a[j][i+n];

C[i] = C[i].add(B[i][j].multiply(A[i][j]));

}

}

//ans为行列式的值

///B为伴随矩阵。若不乘ans,B即为逆矩阵

}

public static int cal(int x,int y){

BigInteger X = B[x][y];

BigInteger Y = C[x].subtract(B[x][y].multiply(A[x][y]));

BigInteger t = yy.subtract(Y);

//cout.println(t+" "+Y);

if( X.compareTo(new BigInteger("0")) == 0 ){

if( t.compareTo(new BigInteger("0")) != 0 )

return 0;

else{

x++;y++;

cout.println(x+" "+y+" "+0);

return 1;

}

}

if( t.abs().gcd(X.abs()).compareTo(X.abs()) == 0 ){

x++;y++;

cout.println(x+" "+y+" "+t.divide(X));

return 1;

}

return 0;

}

public static void main(String args[]) throws Exception{

n = cin.nextInt();

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

for(int j=0;j<n;j++)

A[i][j] = new BigInteger(cin.next());

yy = new BigInteger(cin.next());

for(int i=0;i<n;i++){

for(int j=0;j<n;j++){

a[i][j] = A[i][j];

a[i][j+n] = BigInteger.valueOf(0);

}

a[i][i+n] = BigInteger.valueOf(1);

}

Gauss();

int flag = 0;

for(int i=0;i<n&&flag == 0;i++)

for(int j=0;j<n&&flag == 0;j++)

if( cal(i,j) == 1 ) flag = 1;

if( flag == 0 ) cout.println("-1");

cout.flush();

}

}

## 集合操作

///最高前缀和

for(int i=0;i<k;i++)

for(int s=0;s<=M;s++)

if( s & (1<<i) )

dp2[s^(1<<i)] += dp2[s];

for(int i=0;i<=M;i++)

dp[i] += dp2[i] , dp2[i] = 0;

///枚举子集

for(int a=s;;a=(a-1)&s){

if(!a) break;

}

gcd优化

const int M = 1000;

int vis[M][M];

int \_\_gcd(int a,int b){

while (b>=M)

{

a=a%b;

swap(a,b);

}

return b?vis[a%b][b]:a;

}

int main(){

// freopen("in.txt","r",stdin);

for(int a=0;a<M;a++)

for(int b=0;b<=a;b++)

if(!b) vis[a][b] = a;

else vis[a][b] = vis[b][a%b];

for(int a=0;a<M;a++)

for(int b=0;b<=a;b++)

vis[b][a] = vis[a][b];

return 0;

}