

ACM模板与例题

王春飞



2017-10-9

中国石油大学（华东）

目录

[AC自动机 3](#_Toc498622087)

[HDU3065 3](#_Toc498622088)

[//POJ2778 AC自动机+矩阵快速幂 状态转移 6](#_Toc498622089)

[/\*HDU5955 2016沈阳 AC自动机+高斯消元 \*/ 9](#_Toc498622090)

[FFT 13](#_Toc498622091)

[/\*\*HDU4609\*/ 13](#_Toc498622092)

[//[SPOJ-TSUM] 15](#_Toc498622093)

[FFT模板 16](#_Toc498622094)

[KD-TREE 18](#_Toc498622095)

[/\*欧几里得距离 hdu2966 最近点\*/ 18](#_Toc498622096)

[/\*欧几里得距离HDU4347 最近k个点\*/ 21](#_Toc498622097)

[/\*曼哈顿距离\*/ 24](#_Toc498622098)

[LCA 27](#_Toc498622099)

[数论 28](#_Toc498622100)

[欧拉线性筛 28](#_Toc498622101)

[/\*O(n)求逆元\*/ 29](#_Toc498622102)

[//米勒罗宾 29](#_Toc498622103)

[/\*原根 51nod1135\*/ 30](#_Toc498622104)

[强连通分量 33](#_Toc498622105)

[划分树 34](#_Toc498622106)

[可持久化线段树 36](#_Toc498622107)

[数位DP 38](#_Toc498622108)

[线段树 41](#_Toc498622109)

[SDOI2017相关分析 45](#_Toc498622110)

[矩形面积并 49](#_Toc498622111)

[费用流 51](#_Toc498622112)

[最大流 53](#_Toc498622113)

[无源汇上下界可行流 53](#_Toc498622114)

[/\*超快最大流\*/ 56](#_Toc498622115)

[/\*邻接表\*/ 59](#_Toc498622116)

[/\*邻接矩阵\*/ 61](#_Toc498622117)

[二分图 63](#_Toc498622118)

[匈牙利算法 63](#_Toc498622119)

[树链剖分 64](#_Toc498622120)

[/\*\*「SDOI2016」游戏\*/ 64](#_Toc498622121)

[/\*\*NOI2015软件包管理器\*/ 69](#_Toc498622122)

[浮点数高斯消元 72](#_Toc498622123)

[模板 72](#_Toc498622124)

[///浮点型高斯消元模板 73](#_Toc498622125)

[Treap 75](#_Toc498622126)

[线段树套Treap 77](#_Toc498622127)

[四边形不等式 84](#_Toc498622128)

[定理 84](#_Toc498622129)

[环形石子归并 85](#_Toc498622130)

# AC自动机

## HDU3065

\*\*/

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=(1LL<<60);

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=1000\*70;

int ans[1010];

struct Trie

{

int next[maxn][128],fail[maxn],num[maxn];

int root,L;

int newnode()

{

for(int i = 0;i < 128;i++)

next[L][i] = -1;

num[L++] = 0;

return L-1;

}

void init()

{

L = 0;

root = newnode();

}

void insert(char buf[],int v)

{

int len = strlen(buf);

int now = root;

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

{

if(next[now][buf[i]] == -1)

next[now][buf[i]] = newnode();

now = next[now][buf[i]];

}

num[now]=v;

}

void build()

{

queue<int>Q;

fail[root] = root;

for(int i = 0;i < 128;i++)

if(next[root][i] == -1)

next[root][i] = root;

else

{

fail[next[root][i]] = root;

Q.push(next[root][i]);

}

while( !Q.empty() )

{

int now = Q.front();

Q.pop();

for(int i = 0;i < 128;i++)

if(next[now][i] == -1)

next[now][i] = next[fail[now]][i];

else

{

fail[next[now][i]]=next[fail[now]][i];

Q.push(next[now][i]);

}

}

}

void query(char buf[])

{

int len = strlen(buf);

int now = root;

int res = 0;

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

{

now = next[now][buf[i]];

int temp = now;

while( temp != root )

{

if(num[temp]>0){

ans[num[temp]]++;

}

temp = fail[temp];

}

}

}

};

char buf[2000005];

char ch[1010][60];

Trie ac;

int main()

{

int n;

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

memset(ans,0,sizeof ans);

ac.init();

rep(i,1,n) {

scanf("%s",ch[i]);

ac.insert(ch[i],i);

}

scanf("%s",buf);

ac.build();

ac.query(buf);

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

if(ans[i]!=0) {

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

}

}

}

return 0;

}

## //POJ2778 AC自动机+矩阵快速幂 状态转移

#include<iostream>

#include<cstdio>

#include<cmath>

#include<vector>

#include<cstring>

#include<queue>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=100000;

const ll inf=(1LL<<60);

const double pi=acos(-1);

//ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=150;

int n,m;

int L;

struct Matrix {

ll mm[150][150];

void init()

{

for(int i = 0; i < L; ++i) mm[i][i]=1;

}

};

Matrix multi(Matrix a,Matrix b)

{

Matrix c;

memset(c.mm,0,sizeof c.mm);

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

for(int j = 0; j < L; ++j) {

for(int k = 0; k < L; ++k) {

c.mm[i][j]=(c.mm[i][j]+a.mm[i][k]\*b.mm[k][j])%mod;

}

}

}

return c;

}

Matrix markov;

struct Trie

{

int next[maxn][4],fail[maxn],num[maxn];

int id[226];

int root,L;

int newnode()

{

for(int i = 0;i < 4;i++)

next[L][i] = -1;

num[L++] = 0;

return L-1;

}

void init()

{

id['A']=0;

id['C']=1;

id['G']=2;

id['T']=3;

L = 0;

root = newnode();

}

void insert(char buf[],int v)

{

int len = strlen(buf);

int now = root;

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

{

if(next[now][id[buf[i]]] == -1)

next[now][id[buf[i]]] = newnode();

now = next[now][id[buf[i]]];

}

num[now]=v;

}

void build()

{

queue<int>Q;

fail[root] = root;

for(int i = 0;i < 4;i++)

if(next[root][i] == -1)

next[root][i] = root;

else

{

fail[next[root][i]] = root;

Q.push(next[root][i]);

}

while( !Q.empty() )

{

int now = Q.front();

if(num[fail[now]]) {

num[now]=1;

}

Q.pop();

for(int i = 0;i < 4;i++)

if(next[now][i] == -1)

next[now][i] = next[fail[now]][i];

else

{

fail[next[now][i]]=next[fail[now]][i];

Q.push(next[now][i]);

}

}

}

void query()

{

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

for(int j = 0; j < 4; ++j) {

if(!num[i]&&!num[next[i][j]]) {

markov.mm[i][next[i][j]]++;

}

}

}

}

};

Trie ac;

char ch[15];

int main()

{

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

ac.init();

rep(i,1,n) {

scanf("%s",ch);

ac.insert(ch,i);

}

ac.build();

ac.query();

L=ac.L;

int temp=m;

Matrix ans;

ans.init();

while(temp) {

if(temp&1) ans=multi(ans,markov);

markov=multi(markov,markov);

temp>>=1;

}

ll anss=0;

rep(i,0,ac.L-1) {

anss=(anss+ans.mm[0][i])%mod;

}

cout<<anss<<endl;

return 0;

}

## /\*HDU5955 2016沈阳 AC自动机+高斯消元 \*/

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

#define lson node<<1,l,mid

#define rson node<<1|1,mid+1,r

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=(1LL<<60);

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=300+10;

const int MAXN=320;

double f[maxn][maxn];

int tot;

int eliminate(int n) {

int mxi;double h;

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

mxi=i;

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

if (fabs(f[j][i])>fabs(f[mxi][i])) mxi=j;

if (fabs(f[mxi][i])<1e-9) return 0;//注释

if (mxi!=i) {

for (int j=i;j<n+1;j++) swap(f[i][j],f[mxi][j]);

}

h=f[i][i];

for (int j=i;j<n+1;j++) f[i][j]/=h;

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

if (j!=i) {

h=-f[j][i]/f[i][i];

for (int k=i;k<n+1;k++) f[j][k]+=h\*f[i][k];

}

}

return 1;

}

int t,n,l,pos[maxn];

struct Trie

{

int next[maxn\*maxn][10],fail[maxn\*maxn],end[maxn\*maxn];

int root,L;

int newnode()

{

for(int i = 0;i < 7;i++)

next[L][i] = -1;

end[L++] = 0;

return L-1;

}

void init()

{

L = 0;

root = newnode();

}

void insert(int v)

{

int now = root,u;

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

{

scanf("%d",&u);

if(next[now][u] == -1)

next[now][u] = newnode();

now = next[now][u];

}

pos[v]=now;

end[now]=v+1;

}

void build()

{

queue<int>Q;

fail[root] = root;

for(int i = 1;i < 7;i++)

if(next[root][i] == -1)

next[root][i] = root;

else

{

fail[next[root][i]] = root;

Q.push(next[root][i]);

}

while( !Q.empty() )

{

int now = Q.front();

Q.pop();

for(int i = 1;i < 7;i++)

if(next[now][i] == -1)

next[now][i] = next[fail[now]][i];

else

{

fail[next[now][i]]=next[fail[now]][i];

Q.push(next[now][i]);

}

}

}

int query(char buf[])

{

int now=root,ans=0;

int len=strlen(buf);

rep(i,0,len-1) {

now=next[now][buf[i]=='T'];

int temp=now;

while(temp!=root) {

ans+=end[temp];

end[temp]=0;

temp=fail[temp];

}

}

return ans;

}

};

Trie ac;

int main()

{

scanf("%d",&t);

while(t--) {

ac.init();

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

rep(i,0,n-1) {

ac.insert(i);

}

memset(f,0,sizeof f);

ac.build();

tot=ac.L;

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

if(!ac.end[i]) {

for(int j = 1; j < 7; ++j) {

f[ac.next[i][j]][i]+=1.0/6;

}

}

f[i][i]-=1.0;

}

f[0][tot]-=1;

eliminate(tot+1);

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

if(i) printf(" ");

printf("%.6f",f[pos[i]][tot]);

}

puts("");

}

return 0;

}

# FFT

## /\*\*HDU4609\*/

const double pi=acos(-1);

const int maxn=140000\*2;

complex<double> x1[maxn];

int n,Min=maxn,Max=-maxn;

int a[maxn];

long long num[maxn],sum[maxn],cnt[maxn];

void FFT(complex<double> x[],int n,int p)

{

//把原来依次排列的数变成叶子中的顺序

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

{

if (i>t) swap(x[i],x[t]);

for (int j=n>>1;(t^=j)<j;j>>=1);

}

for (int m=2;m<=n;m<<=1) //枚举每一层

{

complex<double> wn(cos(p\*2\*pi/m),sin(p\*2\*pi/m));

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

{

complex<double> w(1,0),u;

int k=m>>1;

for (int j=0;j<k;j++,w\*=wn)

{

//蝴蝶操作

u=x[i+j+k]\*w;

x[i+j+k]=x[i+j]-u;

x[i+j]=x[i+j]+u;

}

}

}

complex<double>temp(n,0);

if(p==-1) {

for(int i = 0; i < n; ++i) x[i]=x[i]/temp;

}

}

int main()

{

int t;

scanf("%d",&t);

while(t--) {

scanf("%d",&n);

rep(i,0,n-1) scanf("%d",&a[i]),Max=max(Max,a[i]);

Max++;

memset(cnt,0,sizeof cnt);

rep(i,0,n-1) {

cnt[a[i]]++;

}

sort(a,a+n);

int bit=1;

while(bit<2\*Max) bit<<=1;

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

x1[i]=cnt[i];

}

FFT(x1,bit,1);

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

x1[i]=x1[i]\*x1[i];

}

FFT(x1,bit,-1);

long long ans=0;

rep(i,0,bit) num[i]=(long long)(x1[i].real()+0.5);

rep(i,1,n) num[a[i]+a[i]]--;

rep(i,0,bit) num[i]/=2;

sum[0]=0;

rep(i,1,bit) sum[i]=sum[i-1]+num[i];

rep(i,0,n-1) {

ans+=sum[bit]-sum[a[i]];

ans-=(long long)(n-1-i)\*i;

ans-=n-1;

ans-=(long long)(n-1-i)\*(n-2-i)/2;

}

long long tot=(long long)n\*(n-1)\*(n-2)/6;

printf("%.7f\n",double(ans\*1.0)/tot);

}

return 0;

}

## //[SPOJ-TSUM]

const int maxn=140000\*2;

Complex<double> x1[maxn],x2[maxn],x3[maxn],r[maxn];

Complex<double> three(3,0);

int n,Min=maxn,Max=-maxn;

int a[maxn],cnt[maxn];

void FFT(Complex<double> x[],int n,int p)

{

//把原来依次排列的数变成叶子中的顺序

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

{

if (i>t) swap(x[i],x[t]);

for (int j=n>>1;(t^=j)<j;j>>=1);

}

for (int m=2;m<=n;m<<=1) //枚举每一层

{

Complex<double> wn(cos(p\*2\*pi/m),sin(p\*2\*pi/m));

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

{

Complex<double> w(1,0),u;

int k=m>>1;

for (int j=0;j<k;j++,w\*=wn)

{

//蝴蝶操作

u=x[i+j+k]\*w;

x[i+j+k]=x[i+j]-u;

x[i+j]=x[i+j]+u;

}

}

}

Complex<double>temp(n,0);

if(p==-1) {

for(int i = 0; i < n; ++i) x[i]=x[i]/temp;

}

}

int main()

{

scanf("%d",&n);

rep(i,1,n) scanf("%d",&a[i]),Min=min(Min,a[i]),Max=max(Max,a[i]);

Max\*=3;

rep(i,1,n) a[i]-=Min;

rep(i,1,n) {

cnt[a[i]]++;

}

int bit=1<<int(ceil(log(Max + 0.0) / log(2.0)) + 1);

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

x1[i]=cnt[i];

if(i%2==0) x2[i]=cnt[i/2];

else x2[i]=0;

if(i%3==0) x3[i]=cnt[i/3];

else x3[i]=0;

}

FFT(x1,bit,1);

FFT(x2,bit,1);

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

r[i]=x1[i]\*x1[i]\*x1[i]-x1[i]\*x2[i]\*three;

}

FFT(r,bit,-1);

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

ll res=(ll(r[i].real()+0.5)+2\*ll(x3[i].real()))/6;

if(res>0) {

printf("%d : %lld\n",i+Min\*3,res);

}

}

return 0;

}

## FFT模板

#define pi acos(-1)

#define N 200005

using namespace std;

Complex<double> a[N],b[N],p[N];

int n,c[N];

char s[N];

void FFT(Complex<double> x[],int n,int p)

{

//把原来依次排列的数变成叶子中的顺序

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

{

if (i>t) swap(x[i],x[t]);

for (int j=n>>1;(t^=j)<j;j>>=1);

}

for (int m=2;m<=n;m<<=1) //枚举每一层

{

Complex<double> wn(cos(p\*2\*pi/m),sin(p\*2\*pi/m));

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

{

Complex<double> w(1,0),u;

int k=m>>1;

for (int j=0;j<k;j++,w\*=wn)

{

//蝴蝶操作

u=x[i+j+k]\*w;

x[i+j+k]=x[i+j]-u;

x[i+j]=x[i+j]+u;

}

}

}

}

int main()

{

cin>>n;

scanf("%s",s);

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

a[i]=s[n-i-1]-'0';

scanf("%s",s);

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

b[i]=s[n-i-1]-'0';

//把长度变为2的幂次，方便FFT中的迭代

for (int j=n,i=1;(i>>2)<j;i<<=1)

n=i;

FFT(a,n,1),FFT(b,n,1);

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

p[i]=a[i]\*b[i];

//插值

FFT(p,n,-1);

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

c[i]=p[i].real()/n+0.1;

int len=0;

//进位

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

if (c[i])

len=i,c[i+1]+=c[i]/10,c[i]%=10;

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

printf("%d",c[i]);

return 0;

}

# KD-TREE

## /\*欧几里得距离 hdu2966 最近点\*/

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

#define lson node<<1,l,mid

#define rson node<<1|1,mid+1,r

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=(1LL<<60);

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

#define get\_min(a,b) a = min (a, b)

#define get\_max(a,b) a = max (a, b);

template <class T>

inline bool scan (T &ret) {

char c;

int sgn;

if (c = getchar(), c == EOF) return 0; //EOF

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

sgn = (c == '-') ? -1 : 1;

ret = (c == '-') ? 0 : (c - '0');

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

ret \*= sgn;

return 1;

}

#define maxn 100005

struct node {

int d[2], l, r;//节点的点的坐标 左右孩子

long long Max[2], Min[2];//节点中点x的最值 y的最值

int id;

}tree[maxn<<1], tmp;

int n, m;

int root, cmp\_d;

bool cmp (const node &a, const node &b) {

return a.d[cmp\_d] < b.d[cmp\_d] || (a.d[cmp\_d] == b.d[cmp\_d] &&

a.d[cmp\_d^1] < b.d[cmp\_d^1]);

}

void push\_up (int p, int pp) {

get\_min (tree[p].Min[0], tree[pp].Min[0]);

get\_min (tree[p].Min[1], tree[pp].Min[1]);

get\_max (tree[p].Max[0], tree[pp].Max[0]);

get\_max (tree[p].Max[1], tree[pp].Max[1]);

}

int build\_tree (int l, int r, int D) {

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

tree[mid].l = tree[mid].r = 0;

cmp\_d = D;

nth\_element (tree+l+1, tree+mid+1, tree+1+r, cmp);

//按照cmp把第mid元素放在中间 比他小的放左边 比他大的放右边

tree[mid].Max[0] = tree[mid].Min[0] = tree[mid].d[0];

tree[mid].Max[1] = tree[mid].Min[1] = tree[mid].d[1];

if (l != mid) tree[mid].l = build\_tree (l, mid-1, D^1);

if (r != mid) tree[mid].r = build\_tree (mid+1, r, D^1);

if (tree[mid].l) push\_up (mid, tree[mid].l);

if (tree[mid].r) push\_up (mid, tree[mid].r);

return mid;

}

void insert (int now) {

int D = 0, p = root;

while (1) {

push\_up (p, now);//先更新p节点

if (tree[now].d[D] >= tree[p].d[D]) {

if (!tree[p].r) {

tree[p].r = now;

return ;

}

else p = tree[p].r;

}

else {

if (!tree[p].l) {

tree[p].l = now;

return ;

}

else p = tree[p].l;

}

D ^= 1;

}

return ;

}

#define INF 4e18

#define sqr(a) (a)\*(a)

long long ans, x, y;

long long dis (int p, int x, int y) {//点(x,y)在p的管辖范围内的可能最小值

long long ans = 0;

if (x < tree[p].Min[0]) ans += sqr (tree[p].Min[0]-x);

else if (x > tree[p].Max[0]) ans += sqr (x-tree[p].Max[0]);

if (y < tree[p].Min[1]) ans += sqr (tree[p].Min[1]-y);

else if (y > tree[p].Max[1]) ans += sqr (y-tree[p].Max[1]);

return ans;

}

long long distance (int i, long long x, long long y) {

long long xx = tree[i].d[0]-x, yy = tree[i].d[1]-y;

return (xx\*xx+yy\*yy);

}

void query (int p) {

long long dl = INF, dr = INF, d0;

d0 = distance (p, x, y);//初始答案

if (tree[p].d[0] == x && tree[p].d[1] == y) d0 = INF;//重合

get\_min (ans, d0);

if (tree[p].l) dl = dis (tree[p].l, x, y);

if (tree[p].r) dr = dis (tree[p].r, x, y);

if (dl < dr) {

if (dl < ans) query (tree[p].l);

if (dr < ans) query (tree[p].r);

}

else {

if (dr < ans) query (tree[p].r);

if (dl < ans) query (tree[p].l);

}

}

long long res[maxn];

int main () {

//Open ();

int t; cin >> t;

while (t--) {

scan (n);

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

scan (tree[i].d[0]);

scan (tree[i].d[1]);

tree[i].id = i;

}

root = build\_tree (1, n, 0);

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

ans = INF;

x = tree[i].d[0], y = tree[i].d[1];

query (root);

res[tree[i].id] = ans;

}

for (int i = 1; i <= n; i++) printf ("%lld\n", res[i]);

}

return 0;

}

## /\*欧几里得距离HDU4347 最近k个点\*/

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

#define lson node<<1,l,mid

#define rson node<<1|1,mid+1,r

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=(1LL<<60);

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=50000+10;

int n,k;

#define get\_min(a,b) a = min (a, b)

#define get\_max(a,b) a = max (a, b);

struct node {

int d[7], l, r;//节点的点的坐标 左右孩子

long long Max[7], Min[7];//节点中点x的最值 y的最值

int id;

}tree[maxn<<1], tmp,po[maxn<<1];

int now\_d;

bool cmp(const node &a,const node &b) {

return a.d[now\_d] < b.d[now\_d] || (a.d[now\_d] == b.d[now\_d] &&

a.d[now\_d^1] < b.d[now\_d^1]);

}

void push\_up(int p,int pp) {

rep(i,0,k-1){

get\_min(tree[p].Min[i],tree[pp].Min[i]);

get\_max(tree[p].Max[i],tree[pp].Max[i]);

}

}

int build\_tree(int l,int r,int d) {

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

tree[mid].l=tree[mid].r=0;

now\_d=d;

nth\_element(tree+1+l,tree+1+mid,tree+1+r,cmp);

rep(i,0,k-1) tree[mid].Max[i]=tree[mid].Min[i]=tree[mid].d[i];

if(l!=mid) tree[mid].l=build\_tree(l,mid-1,(d+1)%k);

if(r!=mid) tree[mid].r=build\_tree(mid+1,r,(d+1)%k);

if(tree[mid].l) push\_up(mid,tree[mid].l);

if(tree[mid].r) push\_up(mid,tree[mid].r);

return mid;

}

int q[7];

#define sqr(a) (a)\*(a);

ll dis(int p)

{

ll ans=0;

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

if(q[i]<tree[p].Min[i]) ans+=sqr(tree[p].Min[i]-q[i]);

if(q[i]>tree[p].Max[i]) ans+=sqr(q[i]-tree[p].Max[i]);

}

return ans;

}

ll distance(int p)

{

ll ans=0;

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

ans+=sqr(tree[p].d[i]-q[i]);

}

return ans;

}

bool vis[maxn<<1];

pair<ll,int> ans;

void query(int p){

ll dl=inf,dr=inf,d0=inf;

if(!vis[tree[p].id]) d0=distance(p);

if(d0<ans.fi) ans=mp(d0,tree[p].id);

if(tree[p].l) dl=dis(tree[p].l);

if(tree[p].r) dr=dis(tree[p].r);

if(dl<dr) {

if(dl<ans.fi) query(tree[p].l);

if(dr<ans.fi) query(tree[p].r);

}

else {

if(dr<ans.fi) query(tree[p].r);

if(dl<ans.fi) query(tree[p].l);

}

}

int out[15];

int main(){

int m;

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

memset(vis,0,sizeof vis);

rep(i,1,n) {

rep(j,0,k-1) {

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

}

tree[i].id=i;

po[i]=tree[i];

}

int root=build\_tree(1,n,0);

scanf("%d",&m);

while(m--) {

for(int j = 0; j < k; ++j) scanf("%d",&q[j]);

int cnt;

scanf("%d",&cnt);

for(int j = 0; j < cnt; ++j) {

ans=mp(inf,0);

query(root);

out[j]=ans.se;

vis[ans.se]=1;

}

printf ("the closest %d points are:\n", cnt);

for(int j = 0; j < cnt; ++j) {

vis[out[j]]=0;

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

if(i) printf(" ");

printf("%d",po[out[j]].d[i]);

}

puts("");

}

}

}

return 0;

}

## /\*曼哈顿距离\*/

#define maxn 1000010

#define inf 1000000000

using namespace std;

int n,m,r oot,ans,cur;

int fabs(int x)

{

if (x>=0) return x;

else return -x;

}

struct P

{

int d[2],mx[2],mn[2],lch,rch;

int& operator[](int x) {return d[x];}

friend bool operator<(P x,P y) {return x[cur]<y[cur];}

friend int dis(P x,P y) {return fabs(x[0]-y[0])+fabs(x[1]-y[1]);}

}p[maxn];

struct kdtree

{

P t[maxn],T;

int ans;

void update(int k)

{

int l=t[k].lch,r=t[k].rch;

for (int i=0;i<2;i++)

{

t[k].mn[i]=t[k].mx[i]=t[k][i];

if (l) t[k].mn[i]=min(t[k].mn[i],t[l].mn[i]);

if (r) t[k].mn[i]=min(t[k].mn[i],t[r].mn[i]);

if (l) t[k].mx[i]=max(t[k].mx[i],t[l].mx[i]);

if (r) t[k].mx[i]=max(t[k].mx[i],t[r].mx[i]);

}

}

int build(int l,int r,int now)

{

cur=now;

int mid=(l+r)/2;

nth\_element(p+l,p+mid,p+r+1);

t[mid]=p[mid];

for (int i=0;i<2;i++) t[mid].mn[i]=t[mid].mx[i]=t[mid][i];

if (l<mid) t[mid].lch=build(l,mid-1,now^1);

if (r>mid) t[mid].rch=build(mid+1,r,now^1);

update(mid);

return mid;

}

int getmn(P x)

{

int tmp=0;

for (int i=0;i<2;i++)

{

tmp+=max(T[i]-x.mx[i],0);

tmp+=max(x.mn[i]-T[i],0);

}

return tmp;

}

void querymn(int k)

{

ans=min(ans,dis(t[k],T));

int l=t[k].lch,r=t[k].rch,dl=inf,dr=inf;

if (l) dl=getmn(t[l]);

if (r) dr=getmn(t[r]);

if (dl<dr)

{

if (dl<ans) querymn(l);

if (dr<ans) querymn(r);

}

else

{

if (dr<ans) querymn(r);

if (dl<ans) querymn(l);

}

}

void insert(int k,int now)

{

if (T[now]>=t[k][now])

{

if (t[k].rch) insert(t[k].rch,now^1);

else

{

t[k].rch=++n;t[n]=T;

for (int i=0;i<2;i++) t[n].mx[i]=t[n].mn[i]=t[n][i];

}

}

else

{

if (t[k].lch) insert(t[k].lch,now^1);

else

{

t[k].lch=++n;t[n]=T;

for (int i=0;i<2;i++) t[n].mx[i]=t[n].mn[i]=t[n][i];

}

}

update(k);

}

int query(int x,int y)

{

ans=inf;

T[0]=x;T[1]=y;T.lch=0;T.rch=0;

querymn(root);

return ans;

}

void insert1(int x,int y)

{

T[0]=x;T[1]=y;T.lch=0;T.rch=0;insert(root,0);

}

}kdtree;

# LCA

const int maxn=10005;

int n,quest,a,b,cnt,depth[maxn],fa[maxn][35],m;

int NEXT[maxn<<1],END[maxn<<1],last[maxn];

bool mark[maxn];

inline void \_read(int &x){

char t=getchar();bool sign=true;

while(t<'0'||t>'9')

{if(t=='-')sign=false;t=getchar();}

for(x=0;t>='0'&&t<='9';t=getchar())x=x\*10+t-'0';

if(!sign)x=-x;

}

void insert(int a,int b){

END[++cnt]=b;

NEXT[cnt]=last[a];

last[a]=cnt;

}

void dfs(int v){

int i,j,k;

depth[v]=depth[fa[v][0]]+1;

k=ceil(log(depth[v])/log(2));//k为当前讨论的两个节点的倍增值<span style="font-family:Times New Roman;">上限</span>

for(i=1;i<=k;i++)

fa[v][i]=fa[fa[v][i-1]][i-1];//递推计算fa数组

j=last[v];

while(j){

dfs(END[j]);

j=NEXT[j];

}

}

int lca(int x,int y)

{

int i,k,s;

s=ceil(log(n)/log(2));

if(depth[x]<depth[y])swap(x,y);

k=depth[x]-depth[y];

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

if(k&(1<<i))x=fa[x][i];//go\_up操作

if(x==y)return x;

s=ceil(log(depth[x])/log(2));

for(i=s;i>=0;i--)

if(fa[x][i]!=fa[y][i]){ x=fa[x][i]; y=fa[y][i]; }

return fa[x][0];//最终返回公共祖先节点

}

# 数论

## 欧拉线性筛

bool u[maxn];

int su[maxn];

int sunum;

void euler(int n)

{

memset(u,1,sizeof u);

sunum=1;

u[0]=u[1]=0;

rep(i,2,n) {

if(u[i]) su[sunum++]=i;

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

if(i\*su[j]>n) break;

u[i\*su[j]]=0;

if(i%su[j]==0) break;

}

}

}

## /\*O(n)求逆元\*/

ll inv[N];

void init(){

inv[1] = inv[0] = 1;

for(int i = 2;i < N;i++)

inv[i] = inv[mod%i]\*(mod-mod/i)%mod;

}

/\*C(x,y)\*/

const ll mod=1e9+7;

const int maxn=2e5+10;

ll fac[maxn],inv[maxn];

void init()

{

fac[0]=1;

for(int i = 1; i <= maxn-10; ++i) fac[i]=fac[i-1]\*i%mod;

inv[maxn-10]=powmod(fac[maxn-10],mod-2);

for(int i = maxn-11; i >= 0; --i) inv[i]=inv[i+1]\*(i+1)%mod;

}

ll C(int x,int y)

{

return fac[x]\*inv[y]%mod\*inv[x-y]%mod;

}

## //米勒罗宾

ll qpow(ll a,ll b,ll m) {

ll ans=1;

while(b) {

if(b&1) ans\*=a,ans%=m;

a\*=a;a%=m;b>>=1;

}

return ans;

}

bool MillerRabinTest(ll x,ll n) {

ll y=n-1;

while(!(y&1)) y>>=1;

x=qpow(x,y,n);

while(y<n-1&&x!=1&&x!=n-1) x=(x\*x)%n,y<<=(ll)1;

return x==n-1||y&1==1;

}

bool isprime(ll n) {

if(n==2||n==7||n==61) return 1;

if(n==1||(n&1)==0) return 0;

return MillerRabinTest(2,n)&&MillerRabinTest(7,n)&&MillerRabinTest(61,n);

}

//欧拉线性筛

bool u[maxn],su[maxn];

void euler(int n)

{

int i,j,num=1;

for(i=2;i<=n;++i) {

if(u[i]) su[num++]=i;

for(j=1;j<num;++j) {

if(i\*su[j]>n) break;

u[i\*su[j] ]=0;

if(i%su[j]==0 ) break;

}

}

}

## /\*原根 51nod1135\*/

int MOD, w[2][32];

inline int add(int a, int b){

return a + b - MOD >= 0 ? a + b - MOD : a + b;

}

inline int sub(int a, int b){

return a - b + (a - b < 0 ? MOD : 0);

}

inline int mul(int a, int b){

return (long long)a \* b % MOD;

}

int power(int a, int b){

int ret = 1;

for (int t = a; b; b >>= 1){

if (b & 1)ret = mul(ret, t);

t = mul(t, t);

}

return ret;

}

int cal\_root(int mod)

{

int factor[20], num = 0, s = mod - 1;

MOD = mod--;

for (int i = 2; i \* i <= s; i++){

if (s % i == 0){

factor[num++] = i;

while (s % i == 0)s /= i;

}

}

if (s != 1)factor[num++] = s;

for (int i = 2;; i++){

int j = 0;

for (; j < num && power(i, mod / factor[j]) != 1; j++);

if (j == num)return i;

}

}

int main()

{

int p;

cin>>p;

cout<<cal\_root(p)<<endl;

return 0;

}

//浮点型高斯消元

///浮点型高斯消元模板

const double eps=1e-12;

const int maxm=1000;///m个方程，n个变量

const int maxn=1000;

int m,n;

double a[maxm][maxn+1];///增广矩阵

bool free\_x[maxn];///判断是否是不确定的变元

double x[maxn];///解集

int sign(double x)

{

return (x>eps)-(x<-eps);

}

/\*\*返回值：

-1 无解

0 有且仅有一个解

>=1 有多个解，根据free\_x判断哪些是不确定的解

\*/

int Gauss()

{

int i,j;

int row,col,max\_r;

m=n;///n个方程，n个变量的那种情况

for(row=0,col=0;row<m&&col<n;row++,col++)

{

max\_r=row;

for(i=row+1;i<m;i++)///找到当前列所有行中的最大值(做除法时减小误差)

{

if(sign(fabs(a[i][col])-fabs(a[max\_r][col]))>0)

max\_r=i;

}

if(max\_r!=row)

{

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

swap(a[max\_r][j],a[row][j]);

}

if(sign(a[row][col])==0)///当前列row行以下全为0(包括row行)

{

row--;

continue;

}

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

{

if(sign(a[i][col])==0)

continue;

double tmp=a[i][col]/a[row][col];

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

a[i][j]-=a[row][j]\*tmp;

}

}

for(i=row;i<m;i++)///col=n存在0...0,a的情况,无解

{

if(sign(a[i][col]))

return -1;

}

if(row<n)///存在0...0,0的情况,有多个解,自由变元个数为n-row个

{

for(i=row-1;i>=0;i--)

{

int free\_num=0;///自由变元的个数

int free\_index;///自由变元的序号

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

{

if(sign(a[i][j])!=0&&free\_x[j])

free\_num++,free\_index=j;

}

if(free\_num>1)

continue;///该行中的不确定的变元的个数超过1个,无法求解,它们仍然为不确定的变元

///只有一个不确定的变元free\_index,可以求解出该变元,且该变元是确定的

double tmp=a[i][n];

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

{

if(sign(a[i][j])!=0&&j!=free\_index)

tmp-=a[i][j]\*x[j];

}

x[free\_index]=tmp/a[i][free\_index];

free\_x[free\_index]=false;

}

return n-row;

}

///有且仅有一个解,严格的上三角矩阵(n==m)

for(i=n-1;i>=0;i--)

{

double tmp=a[i][n];

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

if(sign(a[i][j])!=0)

tmp-=a[i][j]\*x[j];

x[i]=tmp/a[i][i];

}

return 0;

}///模板结束

# 强连通分量

const int maxn = 20000 + 10;

vector<int> G[maxn];

int pre[maxn], lowlink[maxn], sccno[maxn], dfs\_clock, scc\_cnt;

stack<int> S;

void dfs(int u) {

pre[u] = lowlink[u] = ++dfs\_clock;

S.push(u);

for(int i = 0; i < G[u].size(); i++) {

int v = G[u][i];

if(!pre[v]) {

dfs(v);

lowlink[u] = min(lowlink[u], lowlink[v]);

} else if(!sccno[v]) {

lowlink[u] = min(lowlink[u], pre[v]);

}

}

if(lowlink[u] == pre[u]) {

scc\_cnt++;

for(;;) {

int x = S.top(); S.pop();

sccno[x] = scc\_cnt;

if(x == u) break;

}

}

}

void find\_scc(int n) {

dfs\_clock = scc\_cnt = 0;

memset(sccno, 0, sizeof(sccno));

memset(pre, 0, sizeof(pre));

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

if(!pre[i]) dfs(i);

}

# 划分树

#include <iostream>

#include<cstdio>

#include<cstring>

#include<cmath>

#include<algorithm>

#include<vector>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=(1LL<<60);

const double pi=acos(-1);

//ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=1e5+10;

int tree[30][maxn],sorted[maxn],toleft[30][maxn];

static inline int Rint() {

struct X {

int dig[256];

X() {

for(int i = '0'; i<= '9'; ++i) dig[i]=1;

dig['-']=1;

}

};

static X fuck;

int s=1,v=0,c;

for(;!fuck.dig[c=getchar()];);

if(c=='-') s=0;

else if(fuck.dig[c]) v=c^48;

for(;fuck.dig[c=getchar()];v=v\*10+(c^48));

return s?v:-v;

}

void build(int l,int r,int dep) {

if(l==r) return;

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

int same=mid-l+1;

rep(i,l,r) if(tree[dep][i]<sorted[mid]) same--;

int lpos=l,rpos=mid+1;

rep(i,l,r) {

if(tree[dep][i]<sorted[mid]) tree[dep+1][lpos++]=tree[dep][i];

else if(tree[dep][i]==sorted[mid]&&same>0){

tree[dep+1][lpos++]=tree[dep][i];same--;

}

else tree[dep+1][rpos++]=tree[dep][i];

toleft[dep][i]=toleft[dep][l-1]+lpos-l;

}

build(l,mid,dep+1);

build(mid+1,r,dep+1);

}

int query(int L,int R,int l,int r,int dep,int k)

{

if(l==r) return tree[dep][l];

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

int cnt=toleft[dep][r]-toleft[dep][l-1];

if(cnt>=k) {

int newl=L+toleft[dep][l-1]-toleft[dep][L-1];

int newr=newl+cnt-1;

return query(L,mid,newl,newr,dep+1,k);

}

else {

int newr=r+toleft[dep][R]-toleft[dep][r];

int newl=newr-(r-l-cnt);

return query(mid+1,R,newl,newr,dep+1,k-cnt);

}

}

int main()

{

int n,m;

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

rep(i,1,n) {

tree[0][i]=Rint();

sorted[i]=tree[0][i];

}

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

build(1,n,0);

while(m--) {

int a,b,c;

a=Rint();b=Rint();c=Rint();

printf("%d\n",query(1,n,a,b,0,c));

}

}

return 0;

}

# 可持久化线段树

/\*

HDU2665 求动态区间第K大

\*/

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=(1LL<<60);

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=2e6+10;

int a[maxn],b[maxn],n,tot,root[maxn],q,l,r,k,link[maxn],t;

int hash\_a[maxn];

struct node

{

int ls,rs,size;

};

node tree[maxn];

void insert(int &now,int l,int r,int x)

{

tree[++tot]=tree[now];now=tot;

tree[now].size++;

if(l==r)return;

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

if(x<=mid)insert(tree[now].ls,l,mid,x);

else insert(tree[now].rs,mid+1,r,x);

}

int query(int nl,int nr,int l,int r,int k)

{

if(l==r)return l;

int size=tree[tree[nr].ls].size-tree[tree[nl].ls].size,mid=(l+r)>>1;

if(size>=k) return query(tree[nl].ls,tree[nr].ls,l,mid,k);

else return query(tree[nl].rs,tree[nr].rs,mid+1,r,k-size);

}

int main()

{

scanf("%d",&t);

rep(j,1,t) {

root[0]=0;tot=0;

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

rep(i,1,n) {

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

link[i]=a[i];

}

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

int sz=unique(link+1,link+n+1)-link-1;

rep(i,1,n) {

root[i]=root[i-1];

int v=lower\_bound(link+1,link+1+sz,a[i])-link;

insert(root[i],1,sz,v);

}

rep(i,1,q) {

scanf("%d%d%d",&l,&r,&k);

printf("%d\n",link[query(root[l-1],root[r],1,sz,k)]);

}

}

return 0;

}

# 数位DP

//注意写前导零那一维

/\*

POJ3252 round number

一个数 二进制下 1和0个数相等 则为round number 求[l,r]之间这种数的个数

\*/

#include <iostream>

#include<cstdio>

#include<cstring>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

using namespace std;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=(1LL<<60);

//const double pi=acos(-1);

//ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=1e5+10;

int dp[40][40][40],num[40];

int dfs(int pos,int x,int y,bool flag,bool temp)

{

if(pos<0) return x>=y;

if(!flag&&~dp[pos][x][y]) return dp[pos][x][y];

int u=flag?num[pos]:1;

int res=0;

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

int xx=x,yy=y;

if(!i&&!temp) xx++;

else if(i) yy++;

res+=dfs(pos-1,xx,yy,flag&&i==u,temp&&i==0);

}

return flag?res:dp[pos][x][y]=res;

}

int cal(int m)

{

int pos=0;

while(m){

num[pos++]=m%2;

m/=2;

}

return dfs(pos-1,0,0,1,1);

}

int main()

{

memset(dp,-1,sizeof dp);

int l,r;

while(~scanf("%d%d",&l,&r)) {

printf("%d\n",cal(r)-cal(l-1));

}

return 0;

}

//855E - Salazar Slytherin's Locket

某Base下 [ l , r ]中 各位数字出现偶数次的数的个数 前导零位

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

#define lson node<<1,l,mid

#define rson node<<1|1,mid+1,r

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=(1LL<<60);

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=1e5+10;

ll dp[11][70][2058][2];

int num[40];

int base;

ll dfs(int pos,int x,bool flag,bool temp)

{

if(pos<0) return x==0&&temp!=1;

if(!flag&&~dp[base][pos][x][temp]) return dp[base][pos][x][temp];

int u=flag?num[pos]:(base-1);

ll res=0;

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

if(temp&&i==0)

res+=dfs(pos-1,x,flag&&i==u,1);

else {

res+=dfs(pos-1,x^(1<<i),flag&&i==u,0);

}

}

return flag?res:dp[base][pos][x][temp]=res;

}

ll cal(ll m)

{

int pos=0;

while(m){

num[pos++]=m%base;

m/=base;

}

return dfs(pos-1,0,1,1);

}

int main()

{

memset(dp,-1,sizeof dp);

int q;

base=2;

// cout<<cal(9);

scanf("%d",&q);

ll l,r;

while(q--) {

cin>>base>>l>>r;

cout<<cal(r)-cal(l-1)<<endl;

}

return 0;

}

# 线段树

#define lson node<<1,l,mid

#define rson node<<1|1,mid+1,r

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

{

tree[node].sum=0;tree[node].cov=-1;

if(l==r) return;

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

build(lson);

build(rson);

}

void pushdown(int node,int l,int r)

{

if(tree[node].cov!=-1) {

tree[node<<1].cov=tree[node<<1|1].cov=tree[node].cov;

int mid=l+r>>1;

tree[node<<1].sum=tree[node].cov\*(mid-l+1);

tree[node<<1|1].sum=tree[node].cov\*(r-mid);

tree[node].cov=-1;

}

}

void update(int node,int l,int r,int L,int R,int x)

{

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

tree[node].cov=x;

tree[node].sum=x\*(r-l+1);

return ;

}

pushdown(node,l,r);

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

if(L<=mid) update(lson,L,R,x);

if(mid<R) update(rson,L,R,x);

tree[node].sum=tree[node<<1].sum+tree[node<<1|1].sum;

}

int query(int node,int l,int r,int L,int R)

{

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

return tree[node].sum;

}

pushdown(node,l,r);

int mid=(l+r)>>1,ans=0;

if(L<=mid) ans+=query(lson,L,R);

if(mid<R) ans+=query(rson,L,R);

return ans;

}

const int maxn=1e5+10;

int r,c,m,ql,qr;

const int INF=1000000000;

int \_min,\_max,\_sum;

struct intervalTree

{

int addv[maxn],setv[maxn],sum[maxn],minv[maxn],maxv[maxn];

void maintain(int node,int l,int r)

{

int lc=node<<1,rc=node<<1|1;

// sum[node]=minv[node]=maxv[node]=0;

if(r>l) {

sum[node]=sum[lc]+sum[rc];

minv[node]=min(minv[lc],minv[rc]);

maxv[node]=max(maxv[lc],maxv[rc]);

}

if(setv[node]>=0){

minv[node]=maxv[node]=setv[node];

sum[node]=setv[node]\*(r-l+1);

}

if(addv[node]) {

minv[node]+=addv[node];

maxv[node]+=addv[node];

sum[node]+=addv[node]\*(r-l+1);

}

}

void pushdown(int node)

{

int lc=node<<1,rc=node<<1|1;

if(setv[node]>=0) {

setv[lc]=setv[rc]=setv[node];

addv[lc]=addv[rc]=0;

setv[node]=-1;

}

if(addv[node]>0) {

addv[lc]+=addv[node];

addv[rc]+=addv[node];

addv[node]=0;

}

}

void updateset(int node,int l,int r,int v)

{

int lc=node<<1,rc=node<<1|1;

if(ql<=l&&qr>=r) {

setv[node]=v;

addv[node]=0;

}

else {

pushdown(node);

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

if(ql<=m) updateset(lc,l,m,v);

else maintain(lc,l,m);

if(qr>m) updateset(rc,m+1,r,v);

else maintain(rc,m+1,r);

}

maintain(node,l,r);

}

void updateadd(int node,int l,int r,int v)

{

int lc=node<<1,rc=node<<1|1;

if(ql<=l&&qr>=r) {

addv[node]+=v;

}

else {

pushdown(node);

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

if(ql<=m) updateadd(lc,l,m,v);

else maintain(lc,l,m);

if(qr>m) updateadd(rc,m+1,r,v);

else maintain(rc,m+1,r);

}

maintain(node,l,r);

}

void query(int node,int l,int r,int add)

{

if(setv[node]>=0) {

int v=add+addv[node]+setv[node];

\_sum+=(min(qr,r)-max(l,ql)+1)\*v;

\_min=min(\_min,v);

\_max=max(\_max,v);

}

else if(ql<=l&&qr>=r) {

\_sum+=sum[node]+add\*(r-l+1);

\_min=min(\_min,minv[node]+add);

\_max=max(\_max,maxv[node]+add);

}

else {

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

if(ql<=m)

query(node<<1,l,m,add+addv[node]);

if(qr>m)

query(node<<1|1,m+1,r,add+addv[node]);

}

}

};

intervalTree tree[25];

int main()

{

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

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

while(scanf("%d%d%d",&r,&c,&m)!=EOF) {

memset(tree,0,sizeof tree);

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

memset(tree[i].setv, -1, sizeof(tree[i].setv));

tree[i].setv[1] = 0;

}

int op,x1,x2,v;

while(m--) {

\_sum=0;

\_min=INF;

\_max=0;

scanf("%d%d%d%d%d",&op,&x1,&ql,&x2,&qr);

//if(x1>x2) swap(x1,x2);

if(op==1||op==2) {

scanf("%d",&v);

if(op==1) {

for(int i = x1;i<=x2;++i) {

tree[i].updateadd(1,1,c,v);

}

}

else {

for(int i = x1;i<=x2;++i) {

tree[i].updateset(1,1,c,v);

}

}

}

else {

for(int i=x1;i<=x2;++i) {

tree[i].query(1,1,c,0);

}

printf("%d %d %d\n",\_sum,\_min,\_max);

}

}

}

return 0;

}

## SDOI2017相关分析

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const double inf=1e12;

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=1e5+10;

double sumx[maxn<<2],sumy[maxn<<2],sumxy[maxn<<2],sumxx[maxn<<2],ax[maxn],ay[maxn];

double qx,qy,qxy,qxx;

double covx[maxn<<2],covy[maxn<<2],setx[maxn<<2],sety[maxn<<2];

double pre[maxn];

void maintain(int node)

{

sumx[node]=sumx[node<<1]+sumx[node<<1|1];

sumy[node]=sumy[node<<1]+sumy[node<<1|1];

sumxx[node]=sumxx[node<<1]+sumxx[node<<1|1];

sumxy[node]=sumxy[node<<1]+sumxy[node<<1|1];

}

void pushdown(int node,int l,int r)

{

int len=r-l+1;

if(setx[node]!=inf&&sety[node]!=inf) {

if(l!=r) {

setx[node<<1]=setx[node<<1|1]=setx[node];

sety[node<<1]=sety[node<<1|1]=sety[node];

}

sumx[node]=len\*setx[node]+(double)len\*(r+l)/2;

sumy[node]=len\*sety[node]+(double)len\*(r+l)/2;

sumxx[node]=len\*setx[node]\*setx[node]+2\*setx[node]\*len\*(r+l)/2+pre[r]-pre[l-1];

sumxy[node]=len\*setx[node]\*sety[node]+(setx[node]+sety[node])\*len\*(r+l)/2+pre[r]-pre[l-1];

setx[node]=sety[node]=inf;

if(l!=r) {

covx[node<<1]=covx[node<<1|1]=inf;

covy[node<<1]=covy[node<<1|1]=inf;

}

}

if(covx[node]!=inf&&covy[node]!=inf) {

if(l!=r) {

if(covx[node<<1]==inf) covx[node<<1]=covy[node<<1]=0;

if(covx[node<<1|1]==inf) covx[node<<1|1]=covy[node<<1|1]=0;

covx[node<<1]+=covx[node];covx[node<<1|1]+=covx[node];covy[node<<1]+=covy[node];covy[node<<1|1]+=covy[node];

}

sumxy[node]+=covy[node]\*sumx[node]+covx[node]\*sumy[node]+covx[node]\*covy[node]\*len;

sumxx[node]+=sumx[node]\*covx[node]\*2+len\*covx[node]\*covx[node];

sumx[node]+=len\*covx[node];sumy[node]+=len\*covy[node];

covx[node]=covy[node]=inf;

}

}

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

{

covx[node]=covy[node]=inf;setx[node]=sety[node]=inf;

if(l==r) {

sumx[node]=ax[l];

sumy[node]=ay[l];

sumxy[node]=ax[l]\*ay[l];

sumxx[node]=ax[l]\*ax[l];

return ;

}

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

build(node<<1,l,mid);

build(node<<1|1,mid+1,r);

maintain(node);

}

void update1(int node,int l,int r,int L,int R,double s,double t)

{

// pushdown(node,l,r);

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

if(covx[node]==inf) covx[node]=0;

if(covy[node]==inf) covy[node]=0;

covx[node]+=s;

covy[node]+=t;

pushdown(node,l,r);

return ;

}

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

pushdown(node,l,r);

if(L<=mid) {

update1(node<<1,l,mid,L,R,s,t);

}else {

if(l!=r) pushdown(node<<1,l,mid);

}

if(mid<R) {

update1(node<<1|1,mid+1,r,L,R,s,t);

}

else if(l!=r) pushdown(node<<1|1,mid+1,r);

maintain(node);

}

void update2(int node,int l,int r,int L,int R,double s,double t)

{

// pushdown(node,l,r);

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

setx[node]=s;

sety[node]=t;

covx[node]=covy[node]=inf;

pushdown(node,l,r);

return ;

}

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

pushdown(node,l,r);

if(L<=mid) {

update2(node<<1,l,mid,L,R,s,t);

}else {

if(l!=r) pushdown(node<<1,l,mid);

}

if(mid<R) {

update2(node<<1|1,mid+1,r,L,R,s,t);

}

else if(l!=r) pushdown(node<<1|1,mid+1,r);

maintain(node);

}

void query(int node,int l,int r,int L,int R)

{

pushdown(node,l,r);

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

qx+=sumx[node];

qy+=sumy[node];

qxx+=sumxx[node];

qxy+=sumxy[node];

return ;

}

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

if(L<=mid) {

query(node<<1,l,mid,L,R);

}

if(mid<R) {

query(node<<1|1,mid+1,r,L,R);

}

}

int n,m,op;

int main()

{

// freopen("relative3.in","r",stdin);

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

int l,r;

double s,t;

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

rep(i,1,n) scanf("%lf",&ax[i]);

rep(i,1,n) scanf("%lf",&ay[i]),pre[i]=pre[i-1]+(double)i\*i;

build(1,1,n);

while(m--) {

scanf("%d",&op);

if(op==1) {

scanf("%d%d",&l,&r);

qxy=qx=qy=qxx=0;

query(1,1,n,l,r);

int len=r-l+1;

// cout<<qx<<endl;

printf("%.10f\n",(qxy-qx\*qy/len) /(qxx-qx\*qx/len));

}

else if(op==2) {

scanf("%d%d%lf%lf",&l,&r,&s,&t);

update1(1,1,n,l,r,s,t);

}

else {

scanf("%d%d%lf%lf",&l,&r,&s,&t);

update2(1,1,n,l,r,s,t);

}

}

return 0;

}

# 矩形面积并

#include<iostream>

#include<cstdio>

#include<cstdlib>

#include<algorithm>

#include<cstring>

#define inf 0x7fffffff

#define ll long long

using namespace std;

int n;

int t[801];

double hash[805],sum[805];

struct data{double x1,x2,y;int f;}a[805];

inline bool operator<(data a,data b){return a.y<b.y;}

int find(double x)

{

int l=1,r=2\*n;

while(l<=r)

{

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

if(hash[mid]<x)l=mid+1;

else if(hash[mid]==x)return mid;

else r=mid-1;

}

}

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

{

if(t[k])sum[k]=hash[r+1]-hash[l];

else if(l==r)sum[k]=0;

else sum[k]=sum[k<<1]+sum[k<<1|1];

}

void update(int k,int l,int r,int x,int y,int f)

{

if(x==l&&y==r)

{t[k]+=f;pushup(k,l,r);return;}

    int mid=(l+r)>>1;

    if(y<=mid)update(k<<1,l,mid,x,y,f);

    else if(x>mid)update(k<<1|1,mid+1,r,x,y,f);

    else

    {

     update(k<<1,l,mid,x,mid,f);

     update(k<<1|1,mid+1,r,mid+1,y,f);

    }

    pushup(k,l,r);

}

int main()

{

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

    {

     if(n==0)break;

     memset(t,0,sizeof(t));

        memset(sum,0,sizeof(sum));

        double x1,y1,x2,y2;

        for(int i=1;i<=n;i++)

        {

            scanf("%lf%lf%lf%lf",&x1,&y1,&x2,&y2);

            a[2\*i-1].x1=a[2\*i].x1=x1;

            a[2\*i-1].x2=a[2\*i].x2=x2;

            a[2\*i-1].y=y1;a[2\*i].y=y2;

            a[2\*i-1].f=1;a[2\*i].f=-1;

            hash[2\*i-1]=x1;hash[2\*i]=x2;

        }

        sort(hash+1,hash+2\*n+1);

        sort(a+1,a+2\*n+1);

        double ans=0;

        for(int i=1;i<=2\*n;i++)

        {

            int l=find(a[i].x1),r=find(a[i].x2)-1;

if(l<=r)update(1,1,2\*n,l,r,a[i].f);

            ans+=sum[1]\*(a[i+1].y-a[i].y);

        }

        printf("%.2lf\n",ans);

    }

return 0;

}

# 费用流

const int maxn=500+10;

const int INF=1e9;

struct Edge {

int from, to, cap, flow;

int cost;

};

inline int Min(int aa,int bb)

{

return aa<bb?aa:bb;

}

struct MCMF {

int n, m, s, t;

vector<Edge> edges;

vector<int> G[maxn];

int inq[maxn]; // 是否在队列中

int d[maxn]; // Bellman-Ford

int p[maxn]; // 上一条弧

int a[maxn]; // 可改进量

void init(int n) {

this->n = n;

for(int i = 0; i < n; i++) G[i].clear();

edges.clear();

}

void AddEdge(int from, int to, int cap, int cost) {

edges.push\_back((Edge){from, to, cap, 0, cost});

edges.push\_back((Edge){to, from, 0, 0, -cost});

m = edges.size();

G[from].push\_back(m-2);

G[to].push\_back(m-1);

}

bool BellmanFord(int s, int t, int& flow,int& cost) {

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

memset(inq, 0, sizeof(inq));

d[s] = 0; inq[s] = 1; p[s] = 0; a[s] = INF;

queue<int> Q;

Q.push(s);

while(!Q.empty()) {

int u = Q.front(); Q.pop();

inq[u] = 0;

int l=G[u].size();

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

Edge& e = edges[G[u][i]];

if(e.cap > e.flow && d[e.to] > d[u] + e.cost) {

d[e.to] = d[u] + e.cost;

p[e.to] = G[u][i];

a[e.to] = Min(a[u], e.cap - e.flow);

if(!inq[e.to]) { Q.push(e.to); inq[e.to] = 1; }

}

}

}

if(d[t] == INF) return false;

cost += d[t]\*a[t];

int u = t;

while(u != s) {

edges[p[u]].flow += a[t];

edges[p[u]^1].flow -= a[t];

u = edges[p[u]].from;

}

return true;

}

// 需要保证初始网络中没有负权圈

int Mincost(int s, int t) {

int cost = 0;

int flow=0;

while(BellmanFord(s, t,flow, cost));

return cost;

}

};

MCMF g;

# 最大流

## 无源汇上下界可行流

sgu194 Reactor Cooling

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

#define lson node<<1,l,mid

#define rson node<<1|1,mid+1,r

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=(1LL<<60);

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=200+10;

const int INF=999999999;

struct Edge {

int from, to, cap, flow;

};

bool operator < (const Edge& a, const Edge& b) {

return a.from < b.from || (a.from == b.from && a.to < b.to);

}

int ma[maxn][maxn];

struct Dinic {

int m,s, t;

vector<Edge> edges; // 边数的两倍

vector<int> G[maxn]; // 邻接表，G[i][j]表示结点i的第j条边在e数组中的序号

bool vis[maxn]; // BFS使用

int d[maxn]; // 从起点到i的距离

int cur[maxn]; // 当前弧指针

void AddEdge(int from, int to, int cap) {

edges.push\_back((Edge){from, to, cap, 0});

edges.push\_back((Edge){to, from, 0, 0});

m = edges.size();

G[from].push\_back(m-2);

G[to].push\_back(m-1);

}

bool BFS() {

memset(vis, 0, sizeof(vis));

queue<int> Q;

Q.push(s);

vis[s] = 1;

d[s] = 0;

while(!Q.empty()) {

int x = Q.front(); Q.pop();

for(int i = 0; i < G[x].size(); i++) {

Edge& e = edges[G[x][i]];

if(!vis[e.to] && e.cap > e.flow) {

vis[e.to] = 1;

d[e.to] = d[x] + 1;

Q.push(e.to);

}

}

}

return vis[t];

}

int DFS(int x, int a) {

if(x == t || a == 0) return a;

int flow = 0, f;

for(int& i = cur[x]; i < G[x].size(); i++) {

Edge& e = edges[G[x][i]];

if(d[x] + 1 == d[e.to] && (f = DFS(e.to, min(a, e.cap-e.flow))) > 0) {

e.flow += f;

edges[G[x][i]^1].flow -= f;

flow += f;

a -= f;

if(a == 0) break;

}

}

return flow;

}

int Maxflow(int s, int t) {

this->s = s; this->t = t;

int flow = 0;

while(BFS()) {

memset(cur, 0, sizeof(cur));

flow += DFS(s, INF);

}

return flow;

}

}ac;

int n,m;

int id[200\*200];

int ans[200\*200];

int main()

{

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

int u,v,l,r,st,ed;

int sum=0;

st=0,ed=n+1;

rep(i,1,m) {

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

// ac.n=

ac.AddEdge(u,v,r-l);

id[i]=ac.m-2;

sum+=l;

ac.AddEdge(st,v,l);

ac.AddEdge(u,ed,l);

ans[i]=l;

}

if(ac.Maxflow(st,ed)!=sum) {

puts("NO");

}

else {

puts("YES");

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

printf("%d\n",ac.edges[id[i]].flow+ans[i]);

}

}

return 0;

}

## /\*超快最大流\*/

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

#define C(t) while(t--)

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const int inf=(1<<30);

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int MAXSIZE=100000020;

const int INF=0x7fffffff;

int bufpos;

char buf[MAXSIZE];

void init(){

#ifdef LOCAL

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

#endif // LOCAL

buf[fread(buf,1,MAXSIZE,stdin)]='\0';

bufpos=0;

}

int readint(){

int val=0;

for(;buf[bufpos]<'0' || buf[bufpos]>'9';bufpos++);

for(;buf[bufpos]>='0' && buf[bufpos]<='9' ;bufpos++)

val=val\*10+buf[bufpos]-'0';

return val;

}

typedef long long ll;

struct edge{

int to,cap,rev;

};

const int maxn=1000001;

const int maxm=8000006;

struct graph{

int n,m;

edge w[maxm];

int fr[maxm];

int num[maxn],cur[maxn],first[maxn];

edge e[maxm];

void init(int n){

this->n=n;

m=0;

}

void addedge(int from,int to,int cap){

w[++m]=(edge){to,cap};

num[from]++,fr[m]=from;

w[++m]=(edge){from,0};

num[to]++,fr[m]=to;

}

void prepare(){

//puts("WTF");

first[1]=1;

for(int i=2;i<=n;i++)

first[i]=first[i-1]+num[i-1];

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

num[i]=first[i+1]-1;

num[n]=m;

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

//printf("%d %d\n",fr[i],cur[fr[i]]);

e[first[fr[i]]+(cur[fr[i]]++)]=w[i];

if (!(i%2)){

e[first[fr[i]]+cur[fr[i]]-1].rev=first[w[i].to]+cur[w[i].to]-1;

e[first[w[i].to]+cur[w[i].to]-1].rev=first[fr[i]]+cur[fr[i]]-1;

}

}

}

int q[maxn];

int dist[maxn];

int t;

bool bfs(int s){

int l=1,r=1;

q[1]=s;

memset(dist,-1,(n+1)\*4);

dist[s]=0;

while(l<=r){

int u=q[l++];

for(int i=first[u];i<=num[u];i++){

int v=e[i].to;

if ((dist[v]!=-1) || (!e[i].cap))

continue;

dist[v]=dist[u]+1;

if (v==t)

return true;

q[++r]=v;

}

}

return dist[t]!=-1;

}

int dfs(int u,int flow){

if (u==t)

return flow;

for(int& i=cur[u];i<=num[u];i++){

int v=e[i].to;

if (!e[i].cap || dist[v]!=dist[u]+1)

continue;

int t=dfs(v,min(flow,e[i].cap));

if (t){

e[i].cap-=t;

e[e[i].rev].cap+=t;

return t;

}

}

return 0;

}

ll dinic(int s,int t){

ll ans=0;

this->t=t;

while(bfs(s)){

int flow;

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

cur[i]=first[i];

while(flow=dfs(s,INF))

ans+=(ll)flow;

}

return ans;

}

}g;

int main(){

init();

int n=readint(),m=readint(),s=readint(),t=readint();

g.init(n);

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

int u=readint(),v=readint(),c=readint();

if (u==v) continue;

g.addedge(u,v,c);

}

//puts("WTF");

g.prepare();

//puts("WTF");

printf("%lld\n",g.dinic(s,t));

}

## /\*邻接表\*/

const int maxn=100+10;

const int INF=999999999;

struct Edge {

int from, to, cap, flow;

};

bool operator < (const Edge& a, const Edge& b) {

return a.from < b.from || (a.from == b.from && a.to < b.to);

}

int ma[maxn][maxn];

struct Dinic {

int n, m, s, t;

vector<Edge> edges; // 边数的两倍

vector<int> G[maxn]; // 邻接表，G[i][j]表示结点i的第j条边在e数组中的序号

bool vis[maxn]; // BFS使用

int d[maxn]; // 从起点到i的距离

int cur[maxn]; // 当前弧指针

void AddEdge(int from, int to, int cap) {

edges.push\_back((Edge){from, to, cap, 0});

edges.push\_back((Edge){to, from, 0, 0});

m = edges.size();

G[from].push\_back(m-2);

G[to].push\_back(m-2);

}

bool BFS() {

memset(vis, 0, sizeof(vis));

queue<int> Q;

Q.push(s);

vis[s] = 1;

d[s] = 0;

while(!Q.empty()) {

int x = Q.front(); Q.pop();

for(int i = 0; i < G[x].size(); i++) {

Edge& e = edges[G[x][i]];

if(!vis[e.to] && e.cap > e.flow) {

vis[e.to] = 1;

d[e.to] = d[x] + 1;

Q.push(e.to);

}

}

}

return vis[t];

}

int DFS(int x, int a) {

if(x == t || a == 0) return a;

int flow = 0, f;

for(int& i = cur[x]; i < G[x].size(); i++) {

Edge& e = edges[G[x][i]];

if(d[x] + 1 == d[e.to] && (f = DFS(e.to, min(a, e.cap-e.flow))) > 0) {

e.flow += f;

edges[G[x][i]^1].flow -= f;

flow += f;

a -= f;

if(a == 0) break;

}

}

return flow;

}

int Maxflow(int s, int t) {

this->s = s; this->t = t;

int flow = 0;

while(BFS()) {

memset(cur, 0, sizeof(cur));

flow += DFS(s, INF);

}

return flow;

}

}ac;

## /\*邻接矩阵\*/

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

const int maxn=100+10;

const int INF=999999999;

int cap[maxn][maxn];

int ma[maxn][maxn];

bool vis[maxn]; // BFS使用

int d[maxn]; // 从起点到i的距离

int cur[maxn]; // 当前弧指针

int s,t,n;

bool BFS() {

memset(vis, 0, sizeof(vis));

queue<int> Q;

Q.push(s);

vis[s] = 1;

d[s] = 0;

while(!Q.empty()) {

int x = Q.front(); Q.pop();

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

if(!vis[i] &&cap[x][i]>0) {

vis[i] = 1;

d[i] = d[x] + 1;

Q.push(i);

}

}

}

return vis[t];

}

int DFS(int x, int a) {

if(x == t || a == 0) return a;

int flow = 0, f;

for(int& i = cur[x]; i <=n ; i++) {

if(d[x] + 1 == d[i] && (f = DFS(i, min(a, cap[x][i]))) > 0) {

cap[x][i]-=f;

cap[i][x]+=f;//反向边

a -= f;

flow+=f;

if(a == 0) break;

}

}

return flow;

}

int Maxflow() {

int flow = 0;

while(BFS()) {

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

flow += DFS(s, INF);

}

return flow;

}

int main()

{

int m,k,u,v,c;

s=1;t=2;

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

memset(cap,0,sizeof cap);

rep(i,1,m) {

scanf("%d%d%d",&u,&v,&c);

cap[u][v]=cap[v][u]=c;

}

int ans=Maxflow();

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

while(k--) {

scanf("%d%d%d",&u,&v,&c);

cap[u][v]+=c;

cap[v][u]+=c;

ans+=Maxflow();

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

}

}

return 0;

}

# 二分图

## 匈牙利算法

const int maxn=555;

const int n=100;

vector<int> g[maxn];

int from[maxn],tot;

bool use[maxn];

bool match(int x) {

for(int i = 0; i < g[x].size(); ++i) {

if(!use[g[x][i] ]) {

use[g[x][i]]=true;

if(from[g[x][i] ]==-1||match(from[g[x][i]])) {

from[g[x][i]]=x;

return true;

}

}

}

return false;

}

int hungary() {

tot=0;

memset(from,255,sizeof from);

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

memset(use,0,sizeof use);

if(match(i)) {

++tot;

}

}

return tot;

}

# 树链剖分

## /\*\*「SDOI2016」游戏\*/

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=123456789123456789ll;

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=1e5+10;

int n,m;

vector<pair<int,ll> >g[maxn];

int dep[maxn],siz[maxn],fa[maxn],son[maxn],top[maxn],id[maxn],pos[maxn];

ll d[maxn];

int tot;

/\*\*线段树相关\*/

struct line{

ll a,b;

line(){}

line(ll a,ll b):a(a),b(b){}

ll cal(ll pos) {

return a\*pos+b;

}

};

struct Node

{

line val;

ll ans;

};

Node tree[maxn<<2];

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

{

tree[node].ans=inf;

tree[node].val=line(0,inf);

if(l==r) return;

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

build(node<<1,l,mid);build(node<<1|1,mid+1,r);

}

double ins(line r1,line r2)

{return (double)(r2.b-r1.b)/(r1.a-r2.a);}

void pushup(int node)

{

tree[node].ans=min(tree[node].ans,min(tree[node<<1].ans,tree[node<<1|1].ans));

}

void insert(int node,int l,int r,line a)

{

ll lv=a.cal(d[pos[l]]),rv=a.cal(d[pos[r]]);

ll lp=tree[node].val.cal(d[pos[l]]),rp=tree[node].val.cal(d[pos[r]]);

if(lv>=lp&&rv>=rp) return;

if(lv<=lp&&rv<=rp) tree[node].val=a;

else {

int mid=l+r>>1;

double x=ins(a,tree[node].val),t=(double)(d[pos[mid]]+d[pos[mid+1]])/2.0;

if(lv>lp)

{

if(x>=t)insert(node<<1|1,mid+1,r,a);

else insert(node<<1,l,mid,tree[node].val),tree[node].val=a;

}

else

{

if(x<=t)insert(node<<1,l,mid,a);

else insert(node<<1|1,mid+1,r,tree[node].val),tree[node].val=a;

}

}

tree[node].ans=min(tree[node].ans,tree[node].val.cal(d[pos[l]]));

tree[node].ans=min(tree[node].ans,tree[node].val.cal(d[pos[r]]));

if(l!=r)

pushup(node);

}

void update(int node,int l,int r,int L,int R,line a) {

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

insert(node,l,r,a);

return;

}

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

if(L<=mid)update(node<<1,l,mid,L,R,a);

if(mid<R) update(node<<1|1,mid+1,r,L,R,a);

pushup(node);

}

ll get(int node,int l,int r,int lq,int rq)

{

l=max(l,lq);r=min(r,rq);

return min(tree[node].val.cal(d[pos[l]]),tree[node].val.cal(d[pos[r]]));

}

ll query(int node,int l,int r,int L,int R)

{

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

return tree[node].ans;

}

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

ll ret=get(node,l,r,L,R);

if(mid>=L)ret=min(ret,query(node<<1,l,mid,L,R));

if(mid<R) ret=min(ret,query(node<<1|1,mid+1,r,L,R));

return ret;

}

/\*\*树链剖分相关\*/

void dfs1(int u,int f,int dd) {

dep[u]=dd;

siz[u]=1;

son[u]=-1;

fa[u]=f;int sz=g[u].size(),v;

rep(i,0,sz-1) {

v=g[u][i].fi;

if(v==f) continue;

d[v]=d[u]+g[u][i].se;

dfs1(v,u,dd+1);

siz[u]+=siz[v];

if(son[u]==-1||siz[son[u]]<siz[v]) {

son[u]=v;

}

}

}

void dfs2(int u,int sp) {

top[u]=sp;

id[u]=++tot;

pos[id[u]]=u;

if(son[u]!=-1) dfs2(son[u],sp);

int sz=g[u].size(),v;

rep(i,0,sz-1) {

v=g[u][i].fi;

if(v==fa[u]||v==son[u]) continue;

dfs2(v,v);

}

}

int lca(int x,int y)

{

while(top[x]!=top[y])

{

if(dep[top[x]]<dep[top[y]])swap(x,y);

x=fa[top[x]];

}

return dep[x]<dep[y] ? x:y;

}

void link\_update(int x,int y,line a)

{

while(top[x]!=top[y])

{

update(1,1,n,id[top[x]],id[x],a);

x=fa[top[x]];

}

update(1,1,n,id[y],id[x],a);

}

ll link\_query(int u, int v) {

ll ans=inf;

while (top[u] != top[v]) {

if (dep[top[u]] < dep[top[v]]) {

swap(u, v);

}

ans = min(query(1,1,n,id[top[u]], id[u]), ans);

u = fa[top[u]];

// tp1 = top[u];

}

// if (u == v) return ans;

if (dep[u] < dep[v]) swap(u, v);

ans = min(query(1,1,n,id[v], id[u]), ans);

return ans;

}

int main()

{

tot=0;

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

int u,v,w;

rep(i,1,n-1) {

scanf("%d%d%d",&u,&v,&w);

g[u].pb(mp(v,w));

g[v].pb(mp(u,w));

}

d[1]=0;

dfs1(1,-1,0);

dfs2(1,1);

build(1,1,n);

int op,s,t,a,b;

line tea,teb;

while(m--) {

scanf("%d",&op);

if(op==1) {

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

int lc=lca(s,t);

tea=line(-a,a\*d[s]+b);

link\_update(s,lc,tea);

tea=line(a,a\*d[s]+b-2\*a\*d[lc]);

link\_update(t,lc,tea);

}

else {

scanf("%d%d",&s,&t);

printf("%lld\n",link\_query(s,t));

}

}

return 0;

}

## /\*\*NOI2015软件包管理器\*/

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

#define lson node<<1,l,mid

#define rson node<<1|1,mid+1,r

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=(1LL<<60);

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=1e5+10;

int n,m,tot;

int dep[maxn],top[maxn],siz[maxn],fa[maxn],son[maxn],id[maxn],pos[maxn];

vector<int>g[maxn];

void dfs1(int u,int f,int d)

{

dep[u]=d;

fa[u]=f;

siz[u]=1;

int sz=g[u].size(),v;

rep(i,0,sz-1) {

v=g[u][i];

if(v==f) continue;

dfs1(v,u,d+1);

if(son[u]==-1||siz[son[u]]<siz[v]) {

son[u]=v;

}

siz[u]+=siz[v];

}

}

void dfs2(int u,int sp)

{

top[u]=sp;

id[u]=++tot;

pos[tot]=u;

if(son[u]!=-1) dfs2(son[u],sp);

int sz=g[u].size(),v;

rep(i,0,sz-1) {

v=g[u][i];

if(v==fa[u]||v==son[u]) continue;

dfs2(v,v);

}

}

struct Node

{

int sum,cov;

};

Node tree[maxn<<2];

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

{

tree[node].sum=0;tree[node].cov=-1;

if(l==r) return;

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

build(lson);

build(rson);

}

void pushdown(int node,int l,int r)

{

if(tree[node].cov!=-1) {

tree[node<<1].cov=tree[node<<1|1].cov=tree[node].cov;

int mid=l+r>>1;

tree[node<<1].sum=tree[node].cov\*(mid-l+1);

tree[node<<1|1].sum=tree[node].cov\*(r-mid);

tree[node].cov=-1;

}

}

void update(int node,int l,int r,int L,int R,int x)

{

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

tree[node].cov=x;

tree[node].sum=x\*(r-l+1);

return ;

}

pushdown(node,l,r);

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

if(L<=mid) update(lson,L,R,x);

if(mid<R) update(rson,L,R,x);

tree[node].sum=tree[node<<1].sum+tree[node<<1|1].sum;

}

int query(int node,int l,int r,int L,int R)

{

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

return tree[node].sum;

}

pushdown(node,l,r);

int mid=(l+r)>>1,ans=0;

if(L<=mid) ans+=query(lson,L,R);

if(mid<R) ans+=query(rson,L,R);

return ans;

}

char ch[15];

int main()

{

int x;

memset(son,-1,sizeof son);

scanf("%d",&n);

rep(i,2,n) {

scanf("%d",&x);

g[x+1].pb(i);

}

dfs1(1,-1,1);

dfs2(1,1);

build(1,1,n);

int m,ans,xx;

scanf("%d",&m);

while(m--) {

scanf("%s%d",ch,&x);

x++;

xx=x;

if(ch[0]=='i') {

ans=0;

while(top[x]!=1) {

ans+=query(1,1,n,id[top[x]],id[x]);

update(1,1,n,id[top[x]],id[x],1);

x=fa[top[x]];

}

if(x!=-1) {

ans+=query(1,1,n,id[ top[x]],id[x]);

update(1,1,n,id[top[x]],id[x],1);

}

printf("%d\n",dep[xx]-ans);

}

else {

ans=query(1,1,n,id[x],id[x]+siz[x]-1);

update(1,1,n,id[x],id[x]+siz[x]-1,0);

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

}

}

return 0;

}

# 浮点数高斯消元

## 模板

double f[maxn][maxn]; //左边n\*n矩阵 n+1列为答案

int eliminate(int n) {

int mxi;double h;

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

mxi=i;

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

if (fabs(f[j][i])>fabs(f[mxi][i])) mxi=j;

if (fabs(f[mxi][i])<1e-9) return 0;//注释

if (mxi!=i) {

for (int j=i;j<n+1;j++) swap(f[i][j],f[mxi][j]);

}

h=f[i][i];

for (int j=i;j<n+1;j++) f[i][j]/=h;

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

if (j!=i) {

h=-f[j][i]/f[i][i];

for (int k=i;k<n+1;k++) f[j][k]+=h\*f[i][k];

}

}

return 1;

}

## ///浮点型高斯消元模板

const double eps=1e-12;

const int maxm=1000;///m个方程，n个变量

const int maxn=1000;

int m,n;

double a[maxm][maxn+1];///增广矩阵

bool free\_x[maxn];///判断是否是不确定的变元

double x[maxn];///解集

int sign(double x)

{

return (x>eps)-(x<-eps);

}

/\*\*返回值：

-1 无解

0 有且仅有一个解

>=1 有多个解，根据free\_x判断哪些是不确定的解

\*/

int Gauss()

{

int i,j;

int row,col,max\_r;

m=n;///n个方程，n个变量的那种情况

for(row=0,col=0;row<m&&col<n;row++,col++)

{

max\_r=row;

for(i=row+1;i<m;i++)///找到当前列所有行中的最大值(做除法时减小误差)

{

if(sign(fabs(a[i][col])-fabs(a[max\_r][col]))>0)

max\_r=i;

}

if(max\_r!=row)

{

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

swap(a[max\_r][j],a[row][j]);

}

if(sign(a[row][col])==0)///当前列row行以下全为0(包括row行)

{

row--;

continue;

}

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

{

if(sign(a[i][col])==0)

continue;

double tmp=a[i][col]/a[row][col];

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

a[i][j]-=a[row][j]\*tmp;

}

}

for(i=row;i<m;i++)///col=n存在0...0,a的情况,无解

{

if(sign(a[i][col]))

return -1;

}

if(row<n)///存在0...0,0的情况,有多个解,自由变元个数为n-row个

{

for(i=row-1;i>=0;i--)

{

int free\_num=0;///自由变元的个数

int free\_index;///自由变元的序号

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

{

if(sign(a[i][j])!=0&&free\_x[j])

free\_num++,free\_index=j;

}

if(free\_num>1)

continue;///该行中的不确定的变元的个数超过1个,无法求解,它们仍然为不确定的变元

///只有一个不确定的变元free\_index,可以求解出该变元,且该变元是确定的

double tmp=a[i][n];

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

{

if(sign(a[i][j])!=0&&j!=free\_index)

tmp-=a[i][j]\*x[j];

}

x[free\_index]=tmp/a[i][free\_index];

free\_x[free\_index]=false;

}

return n-row;

}

///有且仅有一个解,严格的上三角矩阵(n==m)

for(i=n-1;i>=0;i--)

{

double tmp=a[i][n];

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

if(sign(a[i][j])!=0)

tmp-=a[i][j]\*x[j];

x[i]=tmp/a[i][i];

}

return 0;

}///模板结束

# Treap

const int inf=(1<<30);

const int maxn=5e4+10;

struct Node

{

Node \*ch[2];

int r,v,s,num;

Node(int v):v(v){ch[0]=ch[1]=NULL;r=rand();s=1;num=1; }

inline int cmp(int x) const {

if(x==v) return -1;

return x<v?0:1;

}

inline void maintain() {

s=num;

if(ch[0]!=NULL) s+=ch[0]->s;

if(ch[1]!=NULL) s+=ch[1]->s;

}

};

void rotate(Node\* &o,int d) {

Node\* k=o->ch[d^1];o->ch[d^1]=k->ch[d];k->ch[d]=o;

o->maintain();

k->maintain();

o=k;

}

void insert(Node\* &o,int x) {//左小右大

if(o==NULL) {

o=new Node(x);

}

else {

int d=(x<=o->v?0:1 ) ;

if(x==o->v) {

(o->s)++;

o->num++;

}

else {

insert(o->ch[d],x);if(o->ch[d]->r>o->r) rotate(o,d^1);

}

}

o->maintain();

}

void remove(Node\* &o,int x) {

int d=o->cmp(x);

if(d==-1) {

if(o->num==1) {

Node\* u=o;

if(o->ch[0]!=NULL&&o->ch[1]!=NULL) {

int d2=(o->ch[0]->r>o->ch[1]->r?1:0);

rotate(o,d2);

remove(o->ch[d2],x);

}

else {

if(o->ch[0]==NULL) o=o->ch[1];

else o=o->ch[0];

delete u;

}

}

else {

(o->num)--;

}

}

else {

remove(o->ch[d],x);

}

if(o!=NULL) o->maintain();

}

int kth(Node\* o,int k) {//第k小

if(o==NULL||k<=0||k>o->s) return 0;

int s=(o->ch[0]==NULL?0:o->ch[0]->s);

if(k==s+1) return o->v;

else if(k<=s) return kth(o->ch[0],k);

else return kth(o->ch[1],k-s-1);

}

int MAXRank(Node\* o,int x){//upperbound

if(o==NULL) return 0;

int s=(o->ch[0]==NULL?0:o->ch[0]->s);

if(o->v==x) {

return s+(o->num);

}

else if(o->v<x) return MAXRank(o->ch[1],x)+s+(o->num);

else return MAXRank(o->ch[0],x);

}

int MINRank(Node\* o,int x){//lowerbound

if(o==NULL) return 0;

int s=(o->ch[0]==NULL?0:o->ch[0]->s);

if(o->v==x) {

return s+1;

}

else if(o->v<x) return MINRank(o->ch[1],x)+s+(o->num);

else return MINRank(o->ch[0],x);

}

int find\_pre(Node \*o,int x)//x节点前驱

{

if(o==NULL) return -inf;

if(o->v >= x) return find\_pre(o->ch[0],x);

return max(o->v,find\_pre(o->ch[1],x));

}

int find\_suf(Node \*o,int x)//x节点后继

{

if(o==NULL) return inf;

if(o->v <= x) return find\_suf(o->ch[1],x);

return min(o->v,find\_suf(o->ch[0],x));

}

//Node\*root=NULL;

//Insert(root);

# 线段树套Treap

这是一道模板题。

您需要写一种数据结构（可参考题目标题），来维护一个有序数列，其中需要提供以下操作：

1. 查询 x x*x* 在区间内的排名；
2. 查询区间内排名为 k k*k* 的值；
3. 修改某一位置上的数值；
4. 查询 x x*x* 在区间内的前趋（前趋定义为小于 x x*x*，且最大的数）；
5. 查询 x x*x* 在区间内的后继（后继定义为大于 x x*x*，且最小的数）。

#### 输入格式

第一行两个数 n,m n, m*n*,*m*，表示长度为 n n*n* 的有序序列和 m m*m* 个操作。  
第二行有 n n*n* 个数，表示有序序列。

下面有 m m*m* 行，每行第一个数表示操作类型：

1. 之后有三个数 l,r,x l, r, x*l*,*r*,*x* 表示查询 x x*x* 在区间 [l,r] [l, r][*l*,*r*] 的排名；
2. 之后有三个数 l,r,k l, r, k*l*,*r*,*k* 表示查询区间 [l,r] [l, r][*l*,*r*] 内排名为 k k*k* 的数；
3. 之后有两个数 pos,x \mathrm{pos}, xpos,*x* 表示将 pos \mathrm{pos}pos 位置的数修改为 x x*x*；
4. 之后有三个数 l,r,x l, r, x*l*,*r*,*x* 表示查询区间 [l,r] [l, r][*l*,*r*] 内 x x*x* 的前趋；
5. 之后有三个数 l,r,x l, r, x*l*,*r*,*x* 表示查询区间 [l,r] [l, r][*l*,*r*] 内 x x*x* 的后继。
6. #include <bits/stdc++.h>
7. #define rep(i,a,n) for (int i=a;i<=n;i++)
8. #define per(i,a,n) for (int i=n;i>=a;i--)
9. #define pb push\_back
10. #define mp make\_pair
11. #define all(x) (x).begin(),(x).end()
12. #define fi first
13. #define se second
14. #define SZ(x) ((int)(x).size())
15. using namespace std;
16. typedef vector<int> vi;
17. typedef long long ll;
18. typedef pair<int,int> pii;
19. const ll mod=1000000007;
20. const int inf=(1<<30);
21. const double pi=acos(-1);
22. ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}
23. inline void pcas(int ca) {printf("Case %d: ",ca);}
24. const int maxn=5e4+10;
25. struct Node
26. {
27. Node \*ch[2];
28. int r,v,s,num;
29. Node(int v):v(v){ch[0]=ch[1]=NULL;r=rand();s=1;num=1; }
30. inline int cmp(int x) const {
31. if(x==v) return -1;
32. return x<v?0:1;
33. }
34. inline void maintain() {
35. s=num;
36. if(ch[0]!=NULL) s+=ch[0]->s;
37. if(ch[1]!=NULL) s+=ch[1]->s;
38. }
39. };
40. void rotate(Node\* &o,int d) {
41. Node\* k=o->ch[d^1];o->ch[d^1]=k->ch[d];k->ch[d]=o;
42. o->maintain();
43. k->maintain();
44. o=k;
45. }
46. void insert(Node\* &o,int x) {
47. if(o==NULL) {
48. o=new Node(x);
49. }
50. else {
51. int d=(x<=o->v?0:1 ) ;
52. if(x==o->v) {
53. (o->s)++;
54. o->num++;
55. }
56. else {
57. insert(o->ch[d],x);if(o->ch[d]->r>o->r) rotate(o,d^1);
58. }
59. }
60. o->maintain();
61. }
62. void remove(Node\* &o,int x) {
63. int d=o->cmp(x);
64. if(d==-1) {
65. if(o->num==1) {
66. Node\* u=o;
67. if(o->ch[0]!=NULL&&o->ch[1]!=NULL) {
68. int d2=(o->ch[0]->r>o->ch[1]->r?1:0);
69. rotate(o,d2);
70. remove(o->ch[d2],x);
71. }
72. else {
73. if(o->ch[0]==NULL) o=o->ch[1];
74. else o=o->ch[0];
75. delete u;
76. }
77. }
78. else {
79. (o->num)--;
80. }
81. }
82. else {
83. remove(o->ch[d],x);
84. }
85. if(o!=NULL) o->maintain();
86. }
87. int kth(Node\* o,int k) {
88. if(o==NULL||k<=0||k>o->s) return 0;
89. int s=(o->ch[0]==NULL?0:o->ch[0]->s);
90. if(k==s+1) return o->v;
91. else if(k<=s) return kth(o->ch[0],k);
92. else return kth(o->ch[1],k-s-1);
93. }
94. int MAXRank(Node\* o,int x){
95. if(o==NULL) return 0;
96. int s=(o->ch[0]==NULL?0:o->ch[0]->s);
97. if(o->v==x) {
98. return s+(o->num);
99. }
100. else if(o->v<x) return MAXRank(o->ch[1],x)+s+(o->num);
101. else return MAXRank(o->ch[0],x);
102. }
103. int MINRank(Node\* o,int x){
104. if(o==NULL) return 0;
105. int s=(o->ch[0]==NULL?0:o->ch[0]->s);
106. if(o->v==x) {
107. return s+1;
108. }
109. else if(o->v<x) return MINRank(o->ch[1],x)+s+(o->num);
110. else return MINRank(o->ch[0],x);
111. }
112. int find\_pre(Node \*o,int x)
113. {
114. if(o==NULL) return -inf;
115. if(o->v >= x) return find\_pre(o->ch[0],x);
116. return max(o->v,find\_pre(o->ch[1],x));
117. }
118. int find\_suf(Node \*o,int x)
119. {
120. if(o==NULL) return inf;
121. if(o->v <= x) return find\_suf(o->ch[1],x);
122. return min(o->v,find\_suf(o->ch[0],x));
123. }
124. struct Seg\_node {
125. Node\* head;
126. int l,r;
127. };
128. Seg\_node tree[maxn<<2];
129. int n,m;
130. int a[maxn];
131. void build(int node,int l,int r)
132. {
133. tree[node].l=l;
134. tree[node].r=r;
135. if(l==r) {
136. insert(tree[node].head,a[l]);
137. return ;
138. }
139. int mid=l+r>>1;
140. build(node<<1,l,mid);
141. build(node<<1|1,mid+1,r);
142. rep(i,l,r) insert(tree[node].head,a[i]);
143. }
144. int query\_rank(int node,int l,int r,int L,int R,int x)
145. {
146. if(L<=l&&r<=R) {
147. return MAXRank(tree[node].head,x);
148. }
149. else {
150. int m=l+r>>1;
151. int ans=0;
152. if(L<=m) ans+=query\_rank(node<<1,l,m,L,R,x);
153. if(R>m) ans+=query\_rank(node<<1|1,m+1,r,L,R,x);
154. return ans;
155. }
156. }
157. void update(int node,int l,int r,int pos,int x)
158. {
159. if(l==r) {
160. remove(tree[node].head,a[l]);
161. insert(tree[node].head,x);
162. return ;
163. }
164. int mid=l+r>>1;
165. if(pos<=mid) update(node<<1,l,mid,pos,x);
166. else update(node<<1|1,mid+1,r,pos,x);
167. remove(tree[node].head,a[pos]);
168. insert(tree[node].head,x);
169. }
170. int query\_kth(int L,int R,int k)
171. {
172. int l=0,r=inf,mid,ans=-1;
173. while(l<=r) {
174. mid=(l+r)/2;
175. int num=query\_rank(1,1,n,L,R,mid);
176. if(num<k) {
177. l=mid+1;
178. }
179. else if(num>k) {
180. r=mid-1;
181. }
182. else {
183. r=mid-1;
184. ans=mid;
185. }
186. }
187. if(ans==-1) {
188. return l;
189. }
190. return ans;
191. }
192. int seg\_findpre(int node,int l,int r,int L,int R,int x) {
193. if(L<=l&&r<=R) {
194. return find\_pre(tree[node].head,x);
195. }
196. int m=l+r>>1;
197. int ans=-inf,temp;
198. if(L<=m) {
199. temp=seg\_findpre(node<<1,l,m,L,R,x);
200. ans=max(ans,temp);
201. }
202. if(R>m) {
203. temp=seg\_findpre(node<<1|1,m+1,r,L,R,x);
204. ans=max(temp,ans);
205. }
206. return ans;
207. }
208. int seg\_findsuf(int node,int l,int r,int L,int R,int x) {
209. if(L<=l&&r<=R) {
210. return find\_suf(tree[node].head,x);
211. }
212. int m=l+r>>1;
213. int ans=inf,temp;
214. if(L<=m) {
215. temp=seg\_findsuf(node<<1,l,m,L,R,x);
216. ans=min(ans,temp);
217. }
218. if(R>m) {
219. temp=seg\_findsuf(node<<1|1,m+1,r,L,R,x);
220. ans=min(temp,ans);
221. }
222. return ans;
223. }
224. int main()
225. {
226. // freopen("input7.in","r",stdin);
227. //freopen("out.txt","w",stdout);
228. int op,l,r,x,pos;
229. scanf("%d%d",&n,&m);
230. rep(i,1,n) scanf("%d",&a[i]);
231. build(1,1,n);
232. while(m--) {
233. scanf("%d",&op);
234. if(op==1) {
235. scanf("%d%d%d",&l,&r,&x);
236. printf("%d\n",query\_rank(1,1,n,l,r,x-1)+1);
237. }
238. else if(op==2) {
239. scanf("%d%d%d",&l,&r,&x);
240. printf("%d\n",query\_kth(l,r,x));
241. }
242. else if(op==3) {
243. scanf("%d%d",&pos,&x);
244. update(1,1,n,pos,x);
245. a[pos]=x;
246. }
247. else if(op==4) {
248. scanf("%d%d%d",&l,&r,&x);
249. printf("%d\n",seg\_findpre(1,1,n,l,r,x));
250. }
251. else {
252. scanf("%d%d%d",&l,&r,&x);
253. printf("%d\n",seg\_findsuf(1,1,n,l,r,x));
254. }
255. }
256. // system("pause");
257. return 0;
258. }

# 四边形不等式

## 定理

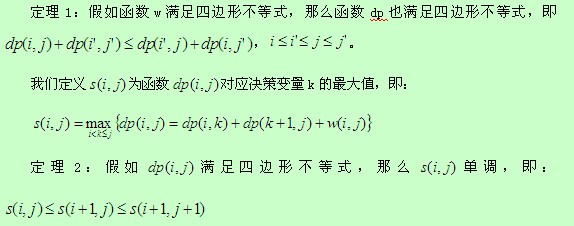
dp(i,j) = min(dp(i,k)+dp(k+1,j)) + w(i,j);(i < j, i<=k<j)

dp(i,j) = MAX;(i>j)

dp(i,j) = 0;(i=j)

上式在动态规划的状态转移方程中是很常见的，对于上式中的w(i,j)  
如果符合w(i`,j) <= w(i,j`) i<i`<j<j`        那么我们称***函数w满足关于区间包含的单调性***  
 如果符合w(i,j)+w(i`,j`) <= w(i`,j)+w(i,j`)    那么我们称***函数w满足四边形不等式***

那么，有两个定理：



显然，由上述定理，本来我们的第三重循环k的范围是由i<=k<j

现在可以缩至s[i][j-1]<=k<=s[i+1][j]，一下就从O(n3)优化至O(n2)了;

## 环形石子归并

N堆石子摆成一个环。现要将石子有次序地合并成一堆。规定每次只能选相邻的2堆石子合并成新的一堆，并将新的一堆石子数记为该次合并的代价。计算将N堆石子合并成一堆的最小代价。

#include <bits/stdc++.h>

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

#define per(i,a,n) for (int i=n;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

#define lson node<<1,l,mid

#define rson node<<1|1,mid+1,r

using namespace std;

typedef vector<int> vi;

typedef long long ll;

typedef pair<int,int> pii;

const ll mod=1000000007;

const ll inf=(1LL<<60);

const double pi=acos(-1);

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

inline void pcas(int ca) {printf("Case %d: ",ca);}

const int maxn=2000+10;

int dp[2010][2010];

int s[maxn][maxn];

int sum[maxn][maxn],a[maxn];

int main()

{

int n;

scanf("%d",&n);

memset(dp,1,sizeof dp);

rep(i,1,n) scanf("%d",&a[i]),a[i+n]=a[i];

rep(i,1,2\*n) {

sum[i][i]=a[i];

s[i][i]=i;

dp[i][i]=0;

rep(j,i+1,2\*n) {

sum[i][j]=sum[i][j-1]+a[j];

}

}

for(int l = 1; l <= 2\*n; ++l) {

for(int st=1;st<=2\*n-l;++st) {

int ed=st+l;

for(int k = s[st][ed-1];k<=s[st+1][ed];++k) {

if(dp[st][ed]>sum[st][ed]+dp[st][k]+dp[k+1][ed]) {

dp[st][ed]=sum[st][ed]+dp[st][k]+dp[k+1][ed];

s[st][ed]=k;

}

}

}

}

int ans=dp[1][n];

for(int i = 2; i <= n; ++i) {

ans=min(ans,dp[i][i+n-1]);

}

cout<<ans<<endl;

return 0;

}