

****

**ACM/ICPC**

沈中海

计算机学院

# Splay

#include <cstdio>

#include <cstring>

#include <cstdlib>

#include <iostream>

#define N 6000000

#define INF 999999999

using namespace std;

long long root, n, m, flag2[N], list[N], fa[N], l[N], r[N], size[N], f[N][3], g[N], flag[N], sum[N], a[N];

char s[20];

long long read(){

long long p=0, q=1;

char ch=getchar();

while (ch<'0' || ch>'9'){

if (ch=='-') q=-1;

ch=getchar();

}

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

return p\*q;

}

void update(long long t){

if (l[t]) fa[l[t]]=t;

if (r[t]) fa[r[t]]=t;

sum[t]=sum[l[t]]+sum[r[t]]+a[t];

size[t]=size[l[t]]+size[r[t]]+1;

f[t][0]=max(f[l[t]][0],sum[l[t]]+f[r[t]][0]+a[t]);

f[t][1]=max(f[r[t]][1],sum[r[t]]+f[l[t]][1]+a[t]);

f[t][2]=f[l[t]][1]+f[r[t]][0]+a[t];

f[t][2]=max(f[t][2],max(f[l[t]][2],f[r[t]][2]));

}

void pushdown(long long t){

if (flag[t]){

if (l[t]) flag[l[t]]^=1;

if (r[t]) flag[r[t]]^=1;

swap(l[t],r[t]);

swap(f[l[t]][0],f[l[t]][1]);

swap(f[r[t]][0],f[r[t]][1]);

flag[t]=0;

}

if (flag2[t]){

if (l[t]){

sum[l[t]]=g[t]\*size[l[t]];

if (g[t]>0)

f[l[t]][0]=f[l[t]][1]=f[l[t]][2]=sum[l[t]];

else

f[l[t]][0]=f[l[t]][1]=0, f[l[t]][2]=g[t];

g[l[t]]=a[l[t]]=g[t];

flag2[l[t]]=1;

}

if (r[t]){

sum[r[t]]=g[t]\*size[r[t]];

if (g[t]>0)

f[r[t]][0]=f[r[t]][1]=f[r[t]][2]=sum[r[t]];

else

f[r[t]][0]=f[r[t]][1]=0, f[r[t]][2]=g[t];

g[r[t]]=a[r[t]]=g[t];

flag2[r[t]]=1;

}

flag2[t]=g[t]=0;

}

}

long long build(long long le, long long ri){

if (le>ri) return 0;

long long mid=le+ri>>1;

l[mid]=build(le,mid-1);

r[mid]=build(mid+1,ri);

update(mid);

return mid;

}

void insert(long long &t, long long k, long long p){

if (!t){

if (!size[t=p]){

size[t]=1;

f[t][0]=f[t][1]=a[t]>0?a[t]:0;

f[t][2]=sum[t]=a[t];

}

return;

}

pushdown(t);

if (size[l[t]]+1<=k) insert(r[t],k-size[l[t]]-1,p);

else insert(l[t],k,p);

update(t);

}

void zig(long long t){

long long f1=fa[t], f2=fa[f1];

if (f2)

if (l[f2]==f1) l[f2]=t;else r[f2]=t;

fa[t]=f2;

l[f1]=r[t];

r[t]=f1;

update(f1);

update(t);

}

void zag(long long t){

long long f1=fa[t], f2=fa[f1];

if (f2)

if (l[f2]==f1) l[f2]=t;else r[f2]=t;

fa[t]=f2;

r[f1]=l[t];

l[t]=f1;

update(f1);

update(t);

}

void splay(long long t){

long long ri=1;

list[1]=t;

for (long long i=1;fa[list[i]];i++) list[++ri]=fa[list[i]];

for (long long i=ri;i;i--) pushdown(list[i]);

long long f1=fa[t], f2=fa[f1];

while (f2){

if (l[f2]==f1)

if (l[f1]==t) zig(f1), zig(t);

else zag(t), zig(t);

else

if (r[f1]==t) zag(f1), zag(t);

else zig(t), zag(t);

f1=fa[t];f2=fa[f1];

}

if (f1)

if (l[f1]==t) zig(t);else zag(t);

root=t;

}

long long find(long long t, long long k){

pushdown(t);

while (size[l[t]]+1!=k){

if (size[l[t]]+1<k)

k-=size[l[t]]+1, t=r[t];

else

t=l[t];

pushdown(t);

}

return t;

}

void del(long long x, long long y){

splay(x);

fa[r[x]]=0;

splay(y);

l[r[root=x]=y]=0;

update(y);

update(x);

}

void modify(long long x, long long y, long long z){

splay(x);

fa[r[x]]=0;

splay(y);

r[root=x]=y;

flag2[l[y]]=1;

g[l[y]]=a[l[y]]=z;

sum[l[y]]=size[l[y]]\*z;

if (z>0)

f[l[y]][0]=f[l[y]][1]=f[l[y]][2]=sum[l[y]];

else

f[l[y]][0]=f[l[y]][1]=0, f[l[y]][2]=z;

update(y);

update(x);

}

void reverse(long long x, long long y){

splay(x);

fa[r[x]]=0;

splay(y);

fa[r[root=x]=y]=x;

flag[l[y]]^=1;

swap(f[l[y]][0],f[l[y]][1]);

update(y);

update(x);

}

void calc(long long x, long long y){

splay(x);

fa[r[x]]=0;

splay(y);

fa[r[root=x]=y]=x;

printf("%d\n", sum[l[y]]);

update(y);

update(x);

}

void print(long long t){

if (!t) return;

pushdown(t);

print(l[t]);

printf("%d ", a[t]);

print(r[t]);

}

int main(){

freopen("sequence4.in","r",stdin);

freopen("1.ans","w",stdout);

n=read();m=read();

for (long long i=1;i<=n;i++) a[i]=read();

f[0][2]=-INF;

root=build(1,n);

a[N-3]=a[N-2]=-INF;

insert(root,0,N-3);

insert(root,n+1,N-2);

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

scanf("%s", s);

if (s[0]=='I'){

long long pos=read(), tot=read(), n2=n+tot, root2;

if (!tot) continue;

for (long long i=n+1;i<=n2;i++) a[i]=read();

root2=build(n+1,n2);

insert(root,pos+1,root2);

splay(root2);

n=n2;

}

if (s[0]=='D'){

long long x=read(), y=read()+x+1;

if (x+1==y) continue;

x=find(root,x);

y=find(root,y);

del(x,y);

}

if (s[2]=='K'){

long long x=read(), y=read()+x+1, z=read();

if (x+1==y) continue;

x=find(root,x);

y=find(root,y);

modify(x,y,z);

}

if (s[0]=='R'){

long long x=read(), y=read()+x+1;

if (x+1==y) continue;

x=find(root,x);

y=find(root,y);

reverse(x,y);

}

if (s[0]=='G'){

long long x=read(), y=read()+x+1;

x=find(root,x);

y=find(root,y);

calc(x,y);

}

if (s[2]=='X'){

printf("%d\n", f[root][2]);

}

}

return 0;

}

# LCT：

#include <cstdio>

#include <cstring>

#include <cstdlib>

#include <iostream>

#define mo 51061

#define N 200000

typedef unsigned int ll;

using namespace std;

int n, q, size[N], l[N], r[N], fa[N], rev[N], list[N];

ll sum[N], f[N], at[N], mt[N];

int read(){

int p=0;

char ch=getchar();

while (ch<'0' || ch>'9') ch=getchar();

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

return p;

}

bool isroot(int t){

return (l[fa[t]]!=t) && (r[fa[t]]!=t);

}

void calc(int u, int m, int a){

if (!u) return;

f[u]=(f[u]\*m+a)%mo;

sum[u]=(sum[u]\*m+a\*size[u])%mo;

at[u]=(at[u]\*m+a)%mo;

mt[u]=(mt[u]\*m)%mo;

}

void update(int t){

if (l[t]) fa[l[t]]=t;

if (r[t]) fa[r[t]]=t;

sum[t]=(f[t]+sum[l[t]]+sum[r[t]])%mo;

size[t]=1+size[l[t]]+size[r[t]];

}

void pushdown(int t){

if (rev[t]){

swap(l[t],r[t]);

if (l[t]) rev[l[t]]^=1;

if (r[t]) rev[r[t]]^=1;

rev[t]=0;

}

int ta=at[t], tm=mt[t];

if (ta || tm!=1){

calc(l[t],tm,ta);

calc(r[t],tm,ta);

}

at[t]=0;mt[t]=1;

}

void zig(int t){

int f1=fa[t], f2=fa[f1];

if (!isroot(f1))

if (l[f2]==f1) l[f2]=t;else r[f2]=t;

fa[t]=f2;

l[f1]=r[t];

r[t]=f1;

update(f1);

update(t);

}

void zag(int t){

int f1=fa[t], f2=fa[f1];

if (!isroot(f1))

if (l[f2]==f1) l[f2]=t;else r[f2]=t;

fa[t]=f2;

r[f1]=l[t];

l[t]=f1;

update(f1);

update(t);

}

void splay(int t){

int ri=1;

list[1]=t;

for (int i=1;!isroot(list[i]);i++) list[++ri]=fa[list[i]];

for (int i=ri;i;i--){

pushdown(list[i]);

}

int f1=fa[t], f2=fa[f1];

while (!isroot(t) && !isroot(f1)){

if (l[f2]==f1)

if (l[f1]==t) zig(f1), zig(t);

else zag(t), zig(t);

else

if (r[f1]==t) zag(f1), zag(t);

else zig(t), zag(t);

f1=fa[t];f2=fa[f1];

}

if (!isroot(t))

if (l[f1]==t) zig(t);else zag(t);

}

void access(int u){

for (int v=0;u;v=u,u=fa[u]){

splay(u);

r[u]=v;

update(u);

}

}

void makeroot(int u){

access(u);

splay(u);

rev[u]^=1;

}

void split(int u, int v){

makeroot(u);

access(v);

splay(v);

}

void link(int u, int v){

makeroot(u);

fa[u]=v;

}

void cut(int u, int v){

split(u,v);

fa[u]=l[v]=0;

update(v);

}

void modify(int u, int v, int m, int a){

split(u,v);

calc(v,m,a);

}

int main(){

n=read();q=read();

for (int i=1;i<=n;i++) size[i]=f[i]=sum[i]=mt[i]=1;

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

int u=read(), v=read();

link(u,v);

}

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

char s[2];

scanf("%s", s);

int u=read(), v=read();

if (s[0]=='+'){

int c=read();

modify(u,v,1,c);

}

if (s[0]=='-'){

cut(u,v);

u=read();v=read();

link(u,v);

}

if (s[0]=='\*'){

int c=read();

modify(u,v,c,0);

}

if (s[0]=='/'){

split(u,v);

printf("%d\n", sum[v]);

}

}

return 0;

}

# AC Automation

#define N 100005

#define C 26

using namespace std;

int n, cnt=1, root=1;

int go[N][C], flag[N], fail[N];

char s[N], st[N];

void init(){

    scanf("%s", st+1);

    cin>>n;

    for (int i=1;i<=n;i++){

        scanf("%s", s+1);

        int len=strlen(s+1);

        for (int j=1, u=root;j<=len;j++){

            int v=s[j]-'a';

            if (!go[u][v]) go[u][v]=++cnt;

            u=go[u][v];

        }

        flag[cnt]=len;

    }

}

int l[N];

void get\_fail(){

    l[1]=root;

    for (int j=0;j<C;j++) go[0][j]=root;

    for (int le=1, ri=1;le<=ri;le++){

        int u=l[le];

        for (int j=0;j<C;j++)

            if (go[u][j])

                l[++ri]=go[u][j],

                fail[go[u][j]]=go[fail[u]][j];

            else go[u][j]=go[fail[u]][j];

    }

}

int top, g[N];

void solve(){

    int len=strlen(st+1);

    g[0]=root;

    int top=1;

    for (int u=root, i=1;i<=len;i++){

        g[top]=go[u][st[i]-'a'];

        s[top]=st[i];

        top-=flag[g[top]];

        u=g[top++];

    }

    s[top]='\0';

    printf("%s\n", s+1);

}

int main(){

    init();

    get\_fail();

    solve();

    return 0;

}

# Suffix array

#include <cstdio>

#include <cstring>

#include <cstdlib>

#include <iostream>

using namespace std;

int ls, a[3000], wv[3000], sa[3000], rk[3000], y[3000], r[3000], h[3000];

char s[3000];

int main(){

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

ls=strlen(s);

int m=max(ls,26);

for (int i=0;i<2\*ls;i++) rk[i]=-1;

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

for (int i=0;i<ls;i++) a[i]=s[i]-'a';

for (int i=0;i<ls;i++) wv[a[i]]++;

for (int i=1;i<m;i++) wv[i]+=wv[i-1];

for (int i=0;i<ls;i++) sa[--wv[a[i]]]=i;

rk[sa[0]]=0;

for (int i=1;i<ls;i++) rk[sa[i]]=rk[sa[i-1]]+(a[sa[i]]!=a[sa[i-1]]);

for (int j=1;j<ls;j\*=2){

int p=0;

for (int i=ls-j;i<ls;i++) y[++p]=i;

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

if (sa[i]>=j) y[++p]=sa[i]-j;

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

for (int i=0;i<ls;i++) wv[rk[i]]++;

for (int i=1;i<m;i++) wv[i]+=wv[i-1];

for (int i=ls;i;i--) sa[--wv[rk[y[i]]]]=y[i];

r[sa[0]]=0;

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

r[sa[i]]=r[sa[i-1]]+(rk[sa[i-1]]!=rk[sa[i]] || rk[j+sa[i-1]]!=rk[j+sa[i]]);

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

rk[i]=r[i];

}

int j=0;

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

if (rk[i]<ls-1){

for (;j+sa[rk[i]+1]<ls && j+i<ls && a[j+sa[rk[i]+1]]==a[i+j];++j);

h[rk[i]]=j?j--:0;

}

for (int i=0;i<ls-1;i++) cout<<h[i]<<endl;

}

return 0;

}

# SAM

#define C 27

#define N 200005

#define MOD 1000000007

#define pii pair<int, int>

typedef long long ll;

using namespace std;

int read(){

    int p=0, q=1;

    char ch=getchar();

    while (ch<'0' || ch>'9') (ch=='-'?q=-1:0), ch=getchar();

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

    return p\*q;

}

int output\_len;

char output[N\*15];

void printi(int x){

    char s[20];

    int ct=0;

    while (x){

        s[ct++]=x%10+'0';

        x/=10;

    }

    for (int i=0;i<ct/2;++i) swap(s[i],s[ct-i-1]);

    s[ct]='\n';

    for (int i=0;i<=ct;i++) output[output\_len++]=s[i];

}

void printll(ll x){

    char s[22];

    int ct=0;

    while (x){

        s[ct++]=x%10+'0';

        x/=10;

    }

    for (int i=0;i<ct/2;++i) swap(s[i],s[ct-i-1]);

    s[ct]='\n';

    for (int i=0;i<=ct;i++) output[output\_len++]=s[i];

}

void add(int &a, int b){

    a+=b;

    if (a>=MOD) a-=MOD;

}

class Sam{

    public:

        int n, cnt, last;

        int maxlen[N], minlen[N];

        int link[N], trans[N][C];

        int in[N], size[N], q[N];

        int f[N];

        char st[N];

        int dp[N];

        int tot[N];

        int flag[N];

        int sg[N];

        ll g[N][C][2];

        friend void printi(int x);

        friend void add(int &a, int b);

        void clear();

        void build(char \*s);

        void extend(int ch);

        ll subst\_diff();

        void get\_size();

        ll get\_sum();

        void lcs();

        void get\_sg(int p);

        void solve();

};

void Sam::clear(){

    last=cnt=1;

    memset(sg,-1,sizeof(sg));

//  memset(f,0,sizeof(f));

//  memset(q,0,sizeof(q));

//  memset(in,0,sizeof(in));

//  memset(size,0,sizeof(size));

//  memset(trans,0,sizeof(trans));

//  memset(minlen,0,sizeof(minlen));

//  memset(maxlen,0,sizeof(maxlen));

//  memset(link,0,sizeof(link));

}

void Sam::build(char \*s){

    clear();

    n=strlen(s);

    for (int i=0;i<n;++i) extend(s[i]-'a');

    for (int i=1;i<=cnt;++i) minlen[i]=maxlen[link[i]]+1;

    get\_size();

}

void Sam::extend(int ch){

    int cur=++cnt;

    int p=last;

    maxlen[cur]=maxlen[p]+1;

    size[cur]=1;

    for (;p && !trans[p][ch];p=link[p]) trans[p][ch]=cur;

    if (!p) link[cur]=1;

    else{

        int q=trans[p][ch];

        if (maxlen[q]==maxlen[p]+1) link[cur]=q;

        else{

            int y=++cnt;

            maxlen[y]=maxlen[p]+1;

            link[y]=link[q];

            size[y]=0;

            memcpy(trans[y], trans[q], sizeof(trans[y]));

            for (;p && trans[p][ch]==q;p=link[p]) trans[p][ch]=y;

            link[q]=link[cur]=y;

        }

    }

    last=cur;

}

void Sam::get\_size(){

    int r=0;

    for (int i=1;i<=cnt;++i) in[link[i]]++;

    for (int i=1;i<=cnt;++i) if (!in[i]) q[++r]=i;

    for (int i=1;i<=r;++i){

        int u=q[i], v=link[u];

        size[v]+=size[u];

        if (!(--in[v])) q[++r]=v;

    }

}

ll Sam::subst\_diff(){

    ll res=0;

    for (int i=1;i<=cnt;++i) res+=maxlen[i]-minlen[i]+1;

    return res;

}

void Sam::lcs(){

    int n=read();

    for (int i=1;i<=n;i++){

        scanf("%s", st+1);

        int len=strlen(st+1);

        int ans=0;

        for (int j=1, p=1, lcs=0;j<=2\*len;j++){

            int ch=st[j>len?j-len:j]-'a';

            while (p>1 && !trans[p][ch]) p=link[p], lcs=maxlen[p];

            if (trans[p][ch]) p=trans[p][ch], ++lcs;

            else lcs=0;

            while (maxlen[link[p]]>=len) p=link[p], lcs=maxlen[p];

            if (lcs>=len && flag[p]!=i) flag[p]=i, ans+=size[p];

        }

        printf("%d\n", ans);

    }

}

void Sam::get\_sg(int p){

    sg[p]=0;

    int flag[C];

    for (int j=0;j<C;j++) flag[j]=0;

    for (int j=0;j<C;j++){

        int v=trans[p][j];

        if (v){

            if (sg[v]==-1) get\_sg(v);

            flag[sg[v]]=1;

            for (int k=0;k<C;k++)

                g[p][k][1]+=g[v][k][1];

        }

    }

    for (int j=0;j<C;j++)

        if (!flag[j]){

            sg[p]=j;

            break;

        }

    ++g[p][sg[p]][1];

    ll sum=0;

    for (int j=0;j<C;j++) sum+=g[p][j][1];

    for (int j=0;j<C;j++) g[p][j][0]=sum-g[p][j][1];

}

void Sam::solve(){

}

Sam sam, sam0, sam1;

char st[N], sta[N], stb[N];

ll K;

void solve(){

    int len0=0, len1=0;

    ll sum=0;

    for (int j=0;j<C;j++){

        sum+=sam0.g[1][j][1]\*sam1.g[1][j][0];

        if (sum>=K) break;

    }

    if (sum<K){

        puts("NO");

        return;

    }

    int s1=1, s2=1, ts1=0, ts2=0;

    while (s1!=ts1 && K>sam1.g[1][sam0.sg[s1]][0]){

        ts1=s1;

        K-=sam1.g[1][sam0.sg[s1]][0];

        for (int j=0;j<C;j++){

            int v=sam0.trans[s1][j];

            if (v){

                ll sum=0;

                for (int k=0;k<C;k++) sum+=sam0.g[v][k][1]\*sam1.g[1][k][0];

                if (sum<K) K-=sum;

                else{

                    sta[len0++]=j+'a';

                    s1=v;

                    break;

                }

            }

        }

    }

    int sg=sam0.sg[s1];

    while (s2!=ts2 && K>(sg!=sam1.sg[s2])){

        ts2=s2;

        K-=sg!=sam1.sg[s2];

        for (int j=0;j<C;j++){

            int v=sam1.trans[s2][j];

            if (v)

                if (sam1.g[v][sg][0]>=K){

                    stb[len1++]=j+'a';

                    s2=v;

                    break;

                }

                else K-=sam1.g[v][sg][0];

        }

    }

    sta[len0]='\0';

    stb[len1]='\0';

    printf("%s\n%s\n", sta, stb);

}

int main(){

    cin>>K;

    scanf("%s", st);

    sam0.build(st);

    sam0.get\_sg(1);

    scanf("%s", st);

    sam1.build(st);

    sam1.get\_sg(1);

    solve();

    return 0;

}

# Miller-Rabin

#include <iostream>

#include <cstdio>

#include <algorithm>

#include <cmath>

#include <cstring>

#include <map>

using namespace std;

const int times = 20;

int number = 0;

map<long long, int>m;

long long Random( long long n ) //生成[ 0 , n ]的随机数

{

return ((double)rand( ) / RAND\_MAX\*n + 0.5);

}

long long q\_mul( long long a, long long b, long long mod ) //快速计算 (a\*b) % mod

{

long long ans = 0;

while(b)

{

if(b & 1)

{

b--;

ans =(ans+ a)%mod;

}

b /= 2;

a = (a + a) % mod;

}

return ans;

}

long long q\_pow( long long a, long long b, long long mod ) //快速计算 (a^b) % mod

{

long long ans = 1;

while(b)

{

if(b & 1)

{

ans = q\_mul( ans, a, mod );

}

b /= 2;

a = q\_mul( a, a, mod );

}

return ans;

}

bool witness( long long a, long long n )//miller\_rabin算法的精华

{//用检验算子a来检验n是不是素数

long long tem = n - 1;

int j = 0;

while(tem % 2 == 0)

{

tem /= 2;

j++;

}

//将n-1拆分为a^r \* s

long long x = q\_pow( a, tem, n ); //得到a^r mod n

if(x == 1 || x == n - 1) return true; //余数为1则为素数

while(j--) //否则试验条件2看是否有满足的 j

{

x = q\_mul( x, x, n );

if(x == n - 1) return true;

}

return false;

}

bool miller\_rabin( long long n ) //检验n是否是素数

{

if(n == 2)

return true;

if(n < 2 || n % 2 == 0)

return false; //如果是2则是素数，如果<2或者是>2的偶数则不是素数

for(int i = 1; i <= times; i++) //做times次随机检验

{

long long a = Random( n - 2 ) + 1; //得到随机检验算子 a

if(!witness( a, n )) //用a检验n是否是素数

return false;

}

return true;

}

int main( )

{

long long tar;

cout<<rand()<<endl;

cout<<RAND\_MAX<<endl;

cout<<Random( 100 - 2 )<<endl;

cout<<Random( 100 - 2 )<<endl;

while(cin >> tar)

{

if(miller\_rabin( tar )) //检验tar是不是素数

cout << "Yes, Prime!" << endl;

else

cout << "No, not prime.." << endl;

}

return 0;

}

# 树链剖分

#define N 31000

#define M 100000

#define INF 999999

typedef long long ll;

using namespace std;

int n, cnt, son[N], sum[N\*4], dep[N], fa[N], f[N\*4], nex[M], nu[M], dfn[N], pre[N], top[N];

char s[10];

int read(){

int p=0, q=1;

char ch=getchar();

while (ch<'0' || ch>'9'){

if (ch=='-') q=-1;

ch=getchar();

}

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

return p\*q;

}

void add(int u, int v){

nex[++cnt]=nex[u];nex[u]=cnt;nu[cnt]=v;

}

void dfs1(int u, int father){

son[u]=1;

int p=0;

for (int j=nex[u];j;j=nex[j]){

int v=nu[j];

if (v==father) continue;

fa[v]=u;

dep[v]=dep[u]+1;

dfs1(v,u);

son[u]+=son[v];

if (son[v]>son[p]) p=v;

}

pre[u]=p;

}

void dfs2(int u, int father){

if (!u) return;

if (pre[father]==u) top[u]=top[father];else top[u]=u;

dfn[u]=++cnt;

dfs2(pre[u],u);

for (int j=nex[u];j;j=nex[j]){

int v=nu[j];

if (v==father || v==pre[u]) continue;

dfs2(v,u);

}

}

void update(int t, int l, int r, int x, int y){

if (l==r){

f[t]=sum[t]=y;

return;

}

int mid=l+r>>1;

if (x<=mid) update(t<<1,l,mid,x,y);else update((t<<1)+1,mid+1,r,x,y);

sum[t]=sum[t<<1]+sum[(t<<1)+1];

f[t]=max(f[t<<1],f[(t<<1)+1]);

}

int get\_max(int t, int l, int r, int le, int ri){

if (le<=l && r<=ri) return f[t];

int mid=l+r>>1, p=-INF;

if (le<=mid) p=max(p,get\_max(t<<1,l,mid,le,ri));

if (ri>mid) p=max(p,get\_max((t<<1)+1,mid+1,r,le,ri));

return p;

}

void query\_max(int u, int v){

int f1=top[u], f2=top[v], ans=-INF;

while (f1!=f2)

if (dep[f1]<dep[f2])

ans=max(ans,get\_max(1,1,n,dfn[f2],dfn[v])),

v=fa[f2],

f2=top[v];

else

ans=max(ans,get\_max(1,1,n,dfn[f1],dfn[u])),

u=fa[f1],

f1=top[u];

ans=max(ans,get\_max(1,1,n,min(dfn[u],dfn[v]),max(dfn[u],dfn[v])));

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

}

int get\_sum(int t, int l, int r, int le ,int ri){

if (le<=l && r<=ri) return sum[t];

int mid=l+r>>1, p=0;

if (le<=mid) p+=get\_sum(t<<1,l,mid,le,ri);

if (ri>mid) p+=get\_sum((t<<1)+1,mid+1,r,le,ri);

return p;

}

void query\_sum(int u, int v){

int f1=top[u], f2=top[v], ans=0;

while (f1!=f2)

if (dep[f1]<dep[f2])

ans+=get\_sum(1,1,n,dfn[f2],dfn[v]),

v=fa[f2],

f2=top[v];

else

ans+=get\_sum(1,1,n,dfn[f1],dfn[u]),

u=fa[f1],

f1=top[u];

ans+=get\_sum(1,1,n,min(dfn[u],dfn[v]),max(dfn[u],dfn[v]));

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

}

int main(){

cnt=n=read();

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

int u=read(), v=read();

add(u,v);

add(v,u);

}

dfs1(1,0);

dfs2(1,cnt=0);

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

update(1,1,n,dfn[i],read());

for (int q=read();q;q--){

scanf("%s", s);

int u=read(), v=read();

if (s[0]=='C') update(1,1,n,dfn[u],v);

if (s[1]=='M') query\_max(u,v);

if (s[1]=='S') query\_sum(u,v);

}

return 0;

}

# Qsort

int n, a[11000];

void qsort(int l, int r){

int i=l, j=r, x=a[l+r>>1];

while (i<=j){

while (a[i]<x && i<r) i++;

while (a[j]>x && j>l) j--;

if (i<=j) swap(a[i++],a[j--]);

}

if (i<r) qsort(i,r);

if (j>l) qsort(l,j);

}

int main(){

srand(unsigned(time(NULL)));

n=300;

for (int i=1;i<=n;i++) a[i]=rand()%100;

qsort(1,n);

for (int i=1;i<=n;i++) cout<<a[i]<<' ';

return 0;

}

# 整体二分

#define N 80010

#define S 2000000

using namespace std;

int n, m, T, x, gt, cnt, DFN, LSH;

int a[N], k[N], u[N], v[N], c[N], ans[N], q[N];

int fa[N][21], dep[N], trans[N\*2], lsh[N\*2], nex[N\*3], nu[N\*3], dfn[N][2];

map<int,int> mp;

char s[S+100];

struct qlz\_ques{

int k, u, v, n;

}l[N\*6], b1[N\*6], b2[N\*6];

int read(){

int p=0;

while (s[x]<'0' || s[x]>'9') x++;

while (s[x]>='0' && s[x]<='9') p=p\*10+s[x++]-'0';

return p;

}

void add\_edge(int u, int v){

nex[++cnt]=nex[u];nex[u]=cnt;nu[cnt]=v;

}

void dfs(int u, int father){

dfn[u][0]=++DFN;

fa[u][0]=father;

for (int i=1;fa[fa[u][i-1]][i-1];i++)

fa[u][i]=fa[fa[u][i-1]][i-1];

//cout<<DFN<<' '<<u<<endl;

for (int j=nex[u];j;j=nex[j]){

int v=nu[j];

if (v==father) continue;

dep[v]=dep[u]+1;

dfs(v,u);

}

dfn[u][1]=DFN+1;

}

int LCA(int u, int v){

if (dep[u]<dep[v]) swap(u,v);

//cout<<u<<' '<<v<<endl;

for (int i=20;i>=0;i--)

if (dep[fa[u][i]]>=dep[v]) u=fa[u][i];

if (u==v) return u;

for (int i=20;i>=0;i--)

if (fa[u][i]!=fa[v][i]) u=fa[u][i], v=fa[v][i];

return fa[u][0];

}

void add(int k, int u, int v){

l[++gt].k=k, l[gt].u=u, l[gt].v=v;

}

void update(int u, int v){

for (int i=u;i<=n;i+=i&(-i)) c[i]+=v;

}

int sum(int u){

int p=0;

for (int i=dfn[u][0];i;i-=i&(-i)) p+=c[i];

return p;

}

void solve(int le, int ri, int L, int R){

//cout<<le<<' '<<ri<<' '<<L<<' '<<R<<endl;

if (le>ri) return;

if (L==R){

for (int i=le;i<=ri;i++)

if (l[i].n) ans[l[i].n]=L;

return;

}

int mid=L+R>>1, ct1=0, ct2=0;

for (int i=le;i<=ri;i++){

if (l[i].n){

int u=l[i].u, v=l[i].v, lca=LCA(u,v),k=sum(u)+sum(v)-sum(lca)-sum(fa[lca][0]);

if (k>=l[i].k)

b2[++ct2]=l[i];

else

l[i].k-=k,

b1[++ct1]=l[i];

}

else

if (l[i].v>mid || l[i].v<-mid)

b2[++ct2]=l[i],

update(l[i].u,l[i].v>0?1:-1);

else

b1[++ct1]=l[i];

}

for (int i=1;i<=ct1;i++) l[le+i-1]=b1[i];

for (int i=1;i<=ct2;i++) l[le+ct1+i-1]=b2[i];

for (int i=le;i<=ri;i++)

if (!l[i].n && (l[i].v>mid || l[i].v<-mid))

update(l[i].u,l[i].v>0?-1:1);

solve(le,le+ct1-1,L,mid);

solve(le+ct1,ri,mid+1,R);

}

int main(){

freopen("network10.in","r",stdin);

freopen("整体二分.out","w",stdout);

//read

fread(s,1,S,stdin);

cnt=n=read();m=read();

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

lsh[++LSH]=a[i]=read();

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

int u=read(), v=read();

add\_edge(u,v);

add\_edge(v,u);

}

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

k[i]=read(),

u[i]=read(),

v[i]=read(),

(!k[i]?lsh[++LSH]=v[i]:0);

//lsh

dfs(dep[1]=1,0);

sort(lsh+1,lsh+1+LSH);

trans[mp[0]=++T]=0;

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

if (lsh[i]!=lsh[i-1]) trans[mp[lsh[i]]=++T]=lsh[i];

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

add(0,dfn[i][0],mp[a[i]]),

add(0,dfn[i][1],-mp[a[i]]);

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

if (k[i])

add(k[i],u[i],v[i]),

l[gt].n=i,

q[i]=1;

else

add(0,dfn[u[i]][0],-mp[a[u[i]]]),

add(0,dfn[u[i]][1],mp[a[u[i]]]),

add(0,dfn[u[i]][0],mp[a[u[i]]=v[i]]),

add(0,dfn[u[i]][1],-mp[v[i]]);

//work

solve(1,gt,0,T);

//int tot=0;

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

if(q[i]){

//tot++;

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

else printf("invalid request!\n");

}

//cout<<n<<' '<<m<<' '<<tot<<' '<<m-tot<<endl;

return 0;

}

# 主席树

#define N 80010

#define M 8001000

#define S 2000000

using namespace std;

int n, m, T, x, cnt, DFN, LSH, ct\_in, ct\_out, cnt\_tree;

int f[M], ls[M], rs[M];

int fa[N][21], dep[N], trans[N\*2], lsh[N\*2], a[N], nex[N\*3], nu[N\*3], root[N], bit[N], b1[N\*2], b2[N\*2], dfn[N][2];

char s[S+100];

map<int,int> mp;

struct qlz\_in{

int n, dfn;

}in[N];

struct qlz\_out{

int n, dfn;

}out[N];

struct qlz\_ques{

int k, u, v;

}l[N];

int read(){

int p=0;

while (s[x]<'0' || s[x]>'9') x++;

while (s[x]>='0' && s[x]<='9') p=p\*10+s[x++]-'0';

return p;

}

void add\_edge(int u, int v){

nex[++cnt]=nex[u];nex[u]=cnt;nu[cnt]=v;

}

bool cmp\_in(qlz\_in a, qlz\_in b){return a.dfn<b.dfn;}

bool cmp\_out(qlz\_out a, qlz\_out b){return a.dfn<b.dfn;}

void dfs(int u, int father){

fa[u][0]=father;

for (int i=1;fa[fa[u][i-1]][i-1];i++)

fa[u][i]=fa[fa[u][i-1]][i-1];

in[++ct\_in].dfn=dfn[u][0]=++DFN;

//cout<<DFN<<' '<<u<<endl;

in[ct\_in].n=u;

for (int j=nex[u];j;j=nex[j]){

int v=nu[j];

if (v==father) continue;

dep[v]=dep[u]+1;

dfs(v,u);

}

out[++ct\_out].dfn=dfn[u][1]=DFN+1;

out[ct\_out].n=u;

}

void add\_b1(int u, int &ct1){

if (root[dfn[u][0]]) b1[++ct1]=root[dfn[u][0]];

for (int i=dfn[u][0];i;i-=i&(-i))

if (bit[i]) b1[++ct1]=bit[i];

}

void add\_b2(int u, int &ct2){

if (root[dfn[u][0]]) b2[++ct2]=root[dfn[u][0]];

for (int i=dfn[u][0];i;i-=i&(-i))

if (bit[i]) b2[++ct2]=bit[i];

}

int LCA(int u, int v){

if (dep[u]<dep[v]) swap(u,v);

//cout<<u<<' '<<v<<endl;

for (int i=20;i>=0;i--)

if (dep[fa[u][i]]>=dep[v]) u=fa[u][i];

if (u==v) return u;

for (int i=20;i>=0;i--)

if (fa[u][i]!=fa[v][i]) u=fa[u][i], v=fa[v][i];

return fa[u][0];

}

void solve(int u, int v, int k){

int ct1=0, ct2=0, l=0, r=T, lca=LCA(u,v);

add\_b1(u,ct1);

add\_b1(v,ct1);

add\_b2(lca,ct2);

add\_b2(fa[lca][0],ct2);

//cout<<u<<' '<<v<<' '<<k<<' '<<lca<<endl;

//for (int i=1;i<=ct1;i++) cout<<b1[i]<<' ';cout<<endl;

//for (int i=1;i<=ct2;i++) cout<<b2[i]<<' ';cout<<endl;

while (l<r){

int mid=l+r>>1, p=0;

for (int i=1;i<=ct1;i++) p+=f[rs[b1[i]]];

for (int i=1;i<=ct2;i++) p-=f[rs[b2[i]]];

//cout<<l<<' '<<r<<' '<<mid<<' '<<p<<' '<<k<<endl;

if (p<k){

for (int i=1;i<=ct1;i++)

b1[i]=ls[b1[i]],

(!b1[i]?b1[i--]=b1[ct1--]:0);

for (int i=1;i<=ct2;i++)

b2[i]=ls[b2[i]],

(!b2[i]?b2[i--]=b2[ct2--]:0);

k-=p;

r=mid;

}

else{

for (int i=1;i<=ct1;i++)

b1[i]=rs[b1[i]],

(!b1[i]?b1[i--]=b1[ct1--]:0);

for (int i=1;i<=ct2;i++)

b2[i]=rs[b2[i]],

(!b2[i]?b2[i--]=b2[ct2--]:0);

l=mid+1;

}

}

if (l) printf("%d\n", trans[l]);

else printf("invalid request!\n");

}

void update(int x, int y, int z){

int ct=0, l=0, r=T;

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

if (!bit[i]) bit[i]=++cnt\_tree;

f[b1[++ct]=bit[i]]+=z;

}

while (l<r){

int mid=l+r>>1;

if (y<=mid){

r=mid;

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

if (!ls[b1[i]]) ls[b1[i]]=++cnt\_tree;

f[b1[i]=ls[b1[i]]]+=z;

}

}

else{

l=mid+1;

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

if (!rs[b1[i]]) rs[b1[i]]=++cnt\_tree;

f[b1[i]=rs[b1[i]]]+=z;

}

}

}

}

int main(){

freopen("network10.in","r",stdin);

freopen("p1146\_主席树静态建树查询优化.out","w",stdout);

//read

fread(s,1,S,stdin);

cnt=n=read();m=read();

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

lsh[++LSH]=a[i]=read();

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

int u=read(), v=read();

add\_edge(u,v);

add\_edge(v,u);

}

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

l[i].k=read(),

l[i].u=read(),

l[i].v=read(),

(!l[i].k?lsh[++LSH]=l[i].v:0);

//lsh

sort(lsh+1,lsh+1+LSH);

trans[mp[0]=++T]=0;

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

if (lsh[i]!=lsh[i-1]) trans[mp[lsh[i]]=++T]=lsh[i];

for (int i=1;i<=n;i++) a[i]=mp[a[i]];

//build

dfs(dep[1]=1,0);

sort(in+1,in+1+n,cmp\_in);

sort(out+1,out+1+n,cmp\_out);

int j=1;

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

int k=root[in[i].dfn]=++cnt\_tree, kk=root[in[i].dfn-1], l=0, r=T, v=a[in[i].n];

while (l<r){

int mid=l+r>>1;

if (v<=mid)

rs[k]=rs[kk],

f[k=ls[k]=++cnt\_tree]=f[kk=ls[kk]]+1,

r=mid;

else

ls[k]=ls[kk],

f[k=rs[k]=++cnt\_tree]=f[kk=rs[kk]]+1,

l=mid+1;

}

while (out[j].dfn==in[i].dfn){

kk=root[in[i].dfn], k=root[in[i].dfn]=++cnt\_tree, l=0, r=T, v=a[out[j++].n];

while (l<r){

int mid=l+r>>1;

if (v<=mid)

rs[k]=rs[kk],

f[k=ls[k]=++cnt\_tree]=f[kk=ls[kk]]-1,

r=mid;

else

ls[k]=ls[kk],

f[k=rs[k]=++cnt\_tree]=f[kk=rs[kk]]-1,

l=mid+1;

}

}

}

//work

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

if (l[i].k)

solve(l[i].u,l[i].v,l[i].k);

else{

int u=l[i].u, v=mp[l[i].v];

update(dfn[u][0],a[u],-1);

update(dfn[u][1],a[u],1);

update(dfn[u][0],a[u]=v,1);

update(dfn[u][1],a[u],-1);

}

return 0;

}

# Cdq(三维偏序)

#define N 600

#define M 500000

using namespace std;

int n, m, x, cnt, ans[M], q[M], c[N][N];

char s[6000010];

struct qlz{

    int n, v, x, y, c, x1, x2, y1, y2;

}l[M], b1[M], b2[M];

inline int read(){

    int p=0;

    while (s[x]<'0' || s[x]>'9') x++;

    while (s[x]>='0' && s[x]<='9') p=p\*10+s[x++]-'0';

    return p;

}

inline bool cmp(qlz a, qlz b){return a.c<b.c;}

inline void update(int x, int y, int z){

    for (int i=x;i<=n;i+=i&(-i))

        for (int j=y;j<=n;j+=j&(-j))

            c[i][j]+=z;

}

inline int sum(int x, int y){

    int p=0;

    for (int i=x;i;i-=i&(-i))

        for (int j=y;j;j-=j&(-j))

            p+=c[i][j];

    return p;

}

inline void solve(int le, int ri, int L, int R){

    if (le>ri) return;

    if (L==R){

        for (int i=le;i<=ri;i++)

            if (!l[i].v) ans[l[i].n]=L;

        return;

    }

    int mid=L+R>>1;

    int ct1=0, ct2=0;

    for (int i=le;i<=ri;i++)

        if (l[i].v){

            if (l[i].v<=mid)

                b1[++ct1]=l[i],

                update(l[i].x,l[i].y,1);

            else

                b2[++ct2]=l[i];

        }

        else{

            int k=sum(l[i].x2,l[i].y2)+sum(l[i].x1-1,l[i].y1-1)-sum(l[i].x1-1,l[i].y2)-sum(l[i].x2,l[i].y1-1);

            if (k>=l[i].c)

                b1[++ct1]=l[i];

            else

                l[i].c-=k,

                b2[++ct2]=l[i];

        }

    for (int i=1;i<=ct1;i++) l[le+i-1]=b1[i];

    for (int i=1;i<=ct2;i++) l[le+ct1+i-1]=b2[i];

    //memcpy(l+le,b1+1,sizeof(l[0])\*ct1);

    //memcpy(l+le+ct1,b2+1,sizeof(l[0])\*ct2);

    for (int i=le;i<=ri;i++)

        if (l[i].v && l[i].v<=mid) update(l[i].x,l[i].y,-1);

    solve(le,le+ct1-1,L,mid);

    solve(le+ct1,ri,mid+1,R);

}

int main(){

    fread(s,1,6000000,stdin);

    n=read();m=read();

    for (int i=1;i<=n;i++)

        for (int j=1;j<=n;j++)

            l[++cnt].c=read(),

            l[cnt].x=i,

            l[cnt].y=j;

    sort(l+1,l+1+cnt,cmp);

    for (int i=1;i<=cnt;i++) q[l[i].v=i]=l[i].c;

    for (int i=1;i<=m;i++)

        l[++cnt].x1=read(),

        l[cnt].y1=read(),

        l[cnt].x2=read(),

        l[cnt].y2=read(),

        l[cnt].c=read(),

        l[cnt].n=i;

    solve(1,cnt,1,n\*n);

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

    return 0;

}

Kmp

#define N 1010000

#define mo 1000000007

typedef long long ll;

using namespace std;

int ls, n, f[N], p[N];

char s[N];

int read(){

    int p=0;

    char ch=getchar();

    while (ch<'0' || ch>'9') ch=getchar();

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

    return p;

}

void pre(){

    ls=strlen(s+1);

    int j=0;

    f[1]=1;

    for (int i=2;i<=ls;i++){

        while (j && s[j+1]!=s[i]) j=p[j];

        f[i]=f[p[i]=j+=s[j+1]==s[i]]+1;

    }

}

void solve(){

    ll ans=1;

    int j=0;

    for (int i=2;i<=ls;i++){

        while (j && s[j+1]!=s[i]) j=p[j];

        if (s[j+1]==s[i]) j++;

        while ((j<<1)>i && j) j=p[j];

        ans=ans\*(f[j]+1)%mo;

    }

    cout<<ans<<endl;

}

void \_\_init(){

    for (int i=read();i;i--){

        scanf("%s", s+1);

        pre();

        solve();

    }

}

int main(){

    \_\_init();

    return 0;

}

点分治：

#define N 100000

using namespace std;

int n, ans, cnt, sum, t[2][3], va[N], nu[N], next[N], son[N], f[N], root;

bool vis[N];

int rd(){

int p=0;

char ch=getchar();

while (ch<'0' || ch>'9') ch=getchar();

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

return p;

}

void add(int u, int v, int w){

next[++cnt]=next[u];next[u]=cnt;nu[cnt]=v;va[cnt]=w;

}

void read(){

cnt=n=rd();

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

int u=rd(), v=rd(), w=rd()%3;

add(u,v,w);

add(v,u,w);

}

}

void getroot(int t, int fa){

son[t]=1;f[t]=0;

for (int j=next[t];j;j=next[j]){

int v=nu[j];

if (vis[v] || v==fa) continue;

getroot(v,t);

son[t]+=son[v];

f[t]=max(son[v],f[t]);

}

f[t]=max(f[t],sum-son[t]);

if (f[t]<f[root]) root=t;

}

void getdeep(int u, int fa, int f){

t[1][f]++;

son[u]=1;

for (int j=next[u];j;j=next[j]){

int v=nu[j];

if (vis[v] || v==fa) continue;

getdeep(v,u,(f+va[j])%3);

son[u]+=son[v];

}

}

void calc(int x, int va){

t[1][0]=t[1][1]=t[1][2]=0;

getdeep(x,0,va);

ans+=t[0][1]\*t[1][2]+t[0][2]\*t[1][1]+t[0][0]\*t[1][0]+t[1][0];

t[0][0]+=t[1][0];

t[0][1]+=t[1][1];

t[0][2]+=t[1][2];

}

void solve(int x){

vis[x]=1;

t[0][0]=t[0][1]=t[0][2]=0;

for (int j=next[x];j;j=next[j]){

int v=nu[j];

if (vis[v]) continue;

calc(v, va[j]);

}

for (int j=next[x];j;j=next[j]){

int v=nu[j];

if (vis[v]) continue;

root=0;sum=son[v];

getroot(v,0);

solve(root);

}

}

int gcd(int a, int b){return !b?a:gcd(b,a%b);}

int main(){

read();

sum=n;

f[0]=n;

getroot(1,0);

solve(root);

ans=ans\*2+n;

int gys=gcd(ans,n\*n);

cout<<ans/gys<<'/'<<n\*n/gys;

return 0;

}

WC2010重建计划---按子树深度递增处理

#define N 300000

#define M 1500000

#define INF (double)99999999\*99999

#define eps 1e-4

using namespace std;

int mx, n, m, L, R, cnt, sum, root, posL, head, tail, check\_flag;

int next[M], nu[M], va[M];

int dep[N], a[N], to[N], q[N], vis[N], flag[N], ff[N], son[N], l[N];

double g[N], f[N];

int read(){

int p=0;

char ch=getchar();

while (ch<'0' || ch>'9') ch=getchar();

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

return p;

}

void add(int u, int v, int w){

next[++cnt]=next[u];next[u]=cnt;nu[cnt]=v;va[cnt]=w;

}

void getroot(int t, int fa){

son[t]=1;

ff[t]=0;

for (int j=next[t];j;j=next[j]){

int v=nu[j];

if (v==fa || vis[v]) continue;

getroot(v,t);

son[t]+=son[v];

ff[t]=max(ff[t],son[v]);

}

ff[t]=max(ff[t],sum-son[t]);

if (ff[t]<ff[root]) root=t;

}

void clear(int t, int fa, int depth){

flag[t]=0;

f[dep[t]=depth]=-INF;

son[t]=1;

for (int j=next[t];j;j=next[j]){

int v=nu[j];

if (v==fa || vis[v]) continue;

clear(v,t,depth+1);

son[t]+=son[v];

dep[t]=max(dep[t],dep[v]);

}

}

void calc(int ll, int rr, double x, int dep){

if (ll>rr) return;

int ri=rr;

while (posL && posL+dep>=L){

while (tail>=head && f[q[tail]]<f[posL]) tail--;

q[++tail]=posL--;

}

while (head<=tail && q[head]+dep>R) head++;

for (int i=ll;i<=rr;i++){

int t=l[i];

flag[t]=1;

if (head<=tail && g[t]+f[q[head]]>=0 || dep>=L && dep<=R && g[t]>=0){

check\_flag=1;

return;

}

for (int j=next[t];j;j=next[j]){

int v=nu[j];

if (vis[v] || flag[v]) continue;

g[v]=g[t]+va[j]-x;

l[++ri]=v;

}

}

calc(rr+1,ri,x,dep+1);

if (check\_flag) return;

for (int i=ll;i<=rr;i++) f[dep]=max(f[dep],g[l[i]]);

}

bool cmp(int x, int y){return dep[x]<dep[y];}

void solve(int t, int la, double x){

clear(t,0,0);

if (dep[t]\*2<L) return;

flag[t]=vis[t]=1;

mx=posL=0;

int ra=la;

for (int j=next[t];j;j=next[j]){

int v=nu[j];

if (vis[v]) continue;

a[ra++]=v;

to[v]=va[j];

}

if (la<ra) sort(a+la,a+ra,cmp);

for (int i=la;i<ra;i++){

int v=a[i];

l[1]=v;

g[v]=to[v]-x;

head=1;tail=0;

calc(1,1,x,1);

posL=mx=max(mx,dep[v]);

if (check\_flag) return;

}

for (int i=la;i<ra;i++){

int v=a[i];

root=0;sum=son[v];

getroot(v,0);

solve(root,ra,x);

}

}

bool check(double x){

check\_flag=0;

for (int i=1;i<=n;i++) vis[i]=0;

sum=ff[root=0]=n;

getroot(1,0);

solve(root,1,x);

return check\_flag;

}

int main(){

freopen("1.in","r",stdin);

freopen("1.out","w",stdout);

cnt=n=read();

L=read();R=read();

double le=0, ri=0, mid;

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

int u=read(), v=read(), w=read();

ri=max(ri,double(w));

add(u,v,w);

add(v,u,w);

}

while (ri-le>eps){

mid=(le+ri)/2;

if (check(mid)) le=mid;

else ri=mid;

}

printf("%.3lf\n", le);

return 0;

}

无向图

1. 桥：low[v]>dfn[u]，则<u,v>为桥

Code:

#include <cstdio>

#include <cstring>

#include <cstdlib>

#include <iostream>

#include <algorithm>

#define N 2010

using namespace std;

int n, m, cnt, ans, dfn[N], low[N], flag[N], bridge[N], nu[N\*3], num[N\*3], nex[N\*3];

int vis[N], x[N], y[N];

int read(){

int p=0;

char ch=getchar();

while (ch<'0' || ch>'9') ch=getchar();

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

return p;

}

void add(int u, int v, int n){

nex[++cnt]=nex[u];nex[u]=cnt;nu[cnt]=v;num[cnt]=n;

}

void initialize(){

cnt=n+n%2+1;

for (int i=1;i<=n;i++) nex[i]=dfn[i]=low[i]=0;

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

}

void tarjan(int u, int from){

dfn[u]=low[u]=++cnt;

for (int j=nex[u];j;j=nex[j])

if (j^from^1){

int v=nu[j];

if (dfn[v]) low[u]=min(low[u],dfn[v]);

else{

tarjan(v,j);

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

bridge[num[j]]=low[v]>dfn[u];

}

}

}

int gcd(int a, int b){return b?gcd(b,a%b):a;}

int main(){

n=read();m=read();

cnt=n+n%2+1;

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

int u=x[i]=read(), v=y[i]=read();

add(u,v,i);

add(v,u,i);

}

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

if (!dfn[i]) tarjan(i,0);

for (int i=1;i<=m;i++) flag[i]=bridge[i];

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

if (!vis[i] && !flag[i]){

initialize();

vis[i]=1;

int tot=1;

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

if (j!=i) add(x[j],y[j],j),add(y[j],x[j],j);

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

if (!dfn[j]) tarjan(j,0);

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

if (bridge[j] && !flag[j]) tot++, vis[j]=1;

ans=gcd(ans,tot);

}

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

if (ans%i==0) printf("%d%c", i, i==ans?'\n':' ');

return 0;

}

1. 割点：对于点u，存在边<u,v>，满足low[v]>=dfn[u]，则u为割点
2. 边双连通分量：分量中无桥边，两种求法

1). Dfs中不走桥边即可。每一个连通分量即是边双连通分量。

2). Dfs找割点，然后对于任意点i和j，如果low[i]==low[j]，那么它们属于同一个边-双连通分量，不会。

1. 点双联通分量：分量中无割点

# 有向图：

1. 桥：同无向图
2. 割点：同无向图
3. 强连通分量：代码如下

#include <cstdio>

#include <cstring>

#include <cstdlib>

#include <iostream>

#define N 200000

#define M 800000

using namespace std;

int cnt, n, m, top, tot, in[N], out[N], f[N], flag[N], dfn[N], low[N], co[N], stack[N], nex[M], nu[M];

int nex2[N], nu2[N];

double ans;

int read(){

int p=0;

char ch=getchar();

while (ch<'0' || ch>'9') ch=getchar();

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

return p;

}

void add(int u, int v){

nex[++cnt]=nex[u];nex[u]=cnt;nu[cnt]=v;

}

void add2(int u, int v){

nex2[++cnt]=nex2[u];nex2[u]=cnt;nu2[cnt]=v;

}

void tarjan(int u){

dfn[u]=low[u]=++cnt;

flag[u]=1;

stack[++top]=u;

for (int j=nex[u];j;j=nex[j]){

int v=nu[j];

if (!dfn[v]){

tarjan(v);

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

}

else if (flag[v]) low[u]=min(low[u],dfn[v]);

}

if (dfn[u]==low[u]){

co[stack[top]]=++tot;

flag[stack[top]]=0;

while (stack[top--]!=u) flag[stack[top]]=0, co[stack[top]]=tot;

}

}

int main(){

cnt=n=read();m=read();

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

int u=read(), v=read();

add(u,v);

}

for (int i=1;i<=n;i++) if (!dfn[i]) tarjan(i);

cnt=n;

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

int u=co[i];

f[u]++;

for (int j=nex[i];j;j=nex[j]){

int v=co[nu[j]];

if (v==u) continue;

in[v]++;

out[u]++;

add2(u,v);

}

}

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

if (!in[i]) ans++;

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

if (!in[i] && f[i]==1){

int flag=1;

for (int j=nex2[i];j;j=nex2[j])

if (in[nu2[j]]==1){

flag=0;

break;

}

if (flag){

ans--;

break;

}

}

ans=(double)(n-ans)/n;

printf("%.6lf\n", ans);

return 0;

}

最大二分图匹配（匈牙利算法）：

#include <cstdio>

#include <cstring>

#include <cstdlib>

#include <iostream>

#include <algorithm>

#define N 110

typedef long long ll;

using namespace std;

int cnt, n, m, g[N][2], fr[N\*2], flag[N\*2], nex[N\*200], nu[N\*200], a[N][N];

ll ans;

int read(){

int p=0;

char ch=getchar();

while (ch<'0' || ch>'9') ch=getchar();

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

return p;

}

void add(int u, int v){

nex[++cnt]=nex[u];nex[u]=cnt;nu[cnt]=v;

}

bool find(int u){

for (int j=nex[u];j;j=nex[j]){

int v=nu[j];

if (flag[v]) continue;

flag[v]=1;

if (!fr[v] || find(fr[v])) {

fr[v]=u;

return 1;

}

}

return 0;

}

int main(){

cnt=(n=read())+(m=read());

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

for (int j=1;j<=m;j++) (a[i][j]=read())?(ans+=a[i][j]-1):0;

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

int ma=0;

for (int j=1;j<=m;j++) ma=max(ma,a[i][j]);

if (ma) ans-=ma-1;

g[i][0]=ma;

}

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

int ma=0;

for (int i=1;i<=n;i++) ma=max(ma,a[i][j]);

if (ma) ans-=ma-1;

g[j][1]=ma;

}

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

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

if (g[i][0]==g[j][1] && a[i][j])

add(i,j+n),

add(j+n,i);

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

memset(flag,0,sizeof(flag));

if (find(i)) ans+=g[i][0]-1;

}

cout<<ans<<endl;

return 0;

}

最小费用流(spfa)

#define N 10000

#define M 50000

#define INF 1000000000

using namespace std;

int n, m ,cnt, s, t, nex[M], nu[M], va[M], w[M];

int dis[N], fl[N], fr[N], flag[N], l[N\*10];

int read(){

int p=0;

char ch=getchar();

while (ch<'0' || ch>'9') ch=getchar();

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

return p;

}

void add(int u, int v, int flow, int cost){

nex[++cnt]=nex[u];nex[u]=cnt;nu[cnt]=v;va[cnt]=flow;w[cnt]=cost;

nex[++cnt]=nex[v];nex[v]=cnt;nu[cnt]=u;va[cnt]=0;w[cnt]=-cost;

}

void \_\_init(){

n=read();m=read();

cnt=(n+1)\*2+1;

s=2\*n+1;t=2\*n+2;

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

int x=read();

add(s,i+n,1,x);

add(s,i,1,0);

add(i+n,t,1,0);

}

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

int u=read(), v=read(), w=read();

if (u>v) swap(u,v);

add(u,v+n,1,w);

}

}

bool spfa(){

int le=0, ri=1;

for (int i=1;i<=t;i++) dis[i]=INF, flag[i]=0;

dis[l[1]=s]=0;

flag[s]=1;

while (le<ri){

int u=l[++le];

flag[u]=0;

for (int j=nex[u];j;j=nex[j]){

if (!va[j]) continue;

int v=nu[j];

if (dis[u]+w[j]<dis[v]){

fr[v]=u;

fl[v]=va[j];

dis[v]=dis[u]+w[j];

if (!flag[v]){

flag[v]=1;

l[++ri]=v;

}

}

}

}

return dis[t]<INF;

}

int sub(){

int j=t, mi=INF;

while (j!=s) mi=min(mi,fl[j]), j=fr[j];

j=t;

while (j!=s){

for (int k=nex[fr[j]];k;k=nex[k])

if (nu[k]==j){

va[k]-=mi;

va[k^1]+=mi;

break;

}

j=fr[j];

}

return mi\*dis[t];

}

void solve(){

int ans=0;

while (spfa()) ans+=sub();

cout<<ans<<endl;

}

int main(){

\_\_init();

solve();

return 0;

}

快速傅里叶变换(FFT)

#include <complex>

#include <iostream>

#include <algorithm>

#define pi acos(-1)

#define N 131077

using namespace std;

typedef long long ll;

typedef long double ld;

typedef complex<double> com;

int n, m, L;

com a[N], b[N];

int c[N], rev[N];

char s[100000];

void init(){

cin>>n;

scanf("%s", s);

for (int i=0;i<n;i++) a[i]=s[n-1-i]-'0';

scanf("%s", s);

for (int i=0;i<n;i++) b[i]=s[n-1-i]-'0';

}

void get\_bit(){

for (n=1, L=0;n<m;n<<=1) L++;

}

void get\_rtable(){

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

rev[i]=(rev[i>>1]>>1)|((i&1)<<(L-1));

}

void mul(com \*a, com\*b){

for (int i=0;i<n;i++) a[i]\*=b[i];

}

void FFT(com \*a, int flag){

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

if (i<rev[i]) swap(a[i],a[rev[i]]);

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

com wn(cos(2\*pi/(i\*2)),flag\*sin(2\*pi/(i\*2)));

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

com w(1,0);

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

com x=a[j+k], y=w\*a[j+k+i];

a[j+k]=x+y;

a[j+k+i]=x-y;

}

}

}

if (flag==-1) for (int i=0;i<n;i++) a[i]/=n;

}

void solve(){

m=n<<1;

get\_bit();

get\_rtable();

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

mul(a,b);

FFT(a,-1);

}

void print(){

for (int i=0;i<m;i++) c[i]=(int)(a[i].real()+0.5);

for (;c[m-1]==0;m--);

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

if (c[i]>=10){

c[i+1]+=c[i]/10;

c[i]%=10;

if (i==m-1) m++;

}

}

for (int i=m-1;i>=0;i--) printf("%d", c[i]);

}

int main(){

init();

solve();

print();

return 0;

}

Isap

#define maxn 80000

#define maxm 3000000

#define inf 2147483647

using namespace std;

struct et

{

    int s,t,val,next;

}e[maxm];

const int dx[4]={0,1,0,-1};

const int dy[4]={1,0,-1,0};

int fir[maxn],dis[maxn],gap[maxn],last[maxn];

int v,s[60][60][60];

int st,ed,n,m,h,num,tot,D,cnt;

int dfs(int now,int flow)

{

    if (now==ed) return flow;

    int sap=0;

    for (int j=last[now];j;j=e[j].next)

    {

        int k=e[j].t;

        if (e[j].val&&dis[now]==dis[k]+1)

        {

            last[now]=j;

            int tmp=dfs(k,min(e[j].val,flow-sap));

            e[j].val-=tmp;

            e[j^1].val+=tmp;

            sap+=tmp;

            if (sap==flow) return sap;

        }

    }

    if (dis[st]>=num) return sap;

    if (!(--gap[dis[now]])) dis[st]=num;

    ++gap[++dis[now]];

    last[now]=fir[now];

    return sap;

}

void add(int x,int y,int z)

{

    e[++tot].s=x; e[tot].t=y; e[tot].val=z; e[tot].next=fir[x]; fir[x]=tot;

    e[++tot].s=y; e[tot].t=x; e[tot].val=0; e[tot].next=fir[y]; fir[y]=tot;

}

int main()

{

    scanf("%d%d%d",&n,&m,&h);

    scanf("%d",&D);

    for (int k=1;k<=h+1;k++)

        for (int i=1;i<=n;i++)

            for (int j=1;j<=m;j++)

                s[k][i][j]=++cnt;

    st=0; ed=cnt+1; num=cnt+2; tot=1;

    for (int i=1;i<=n;i++)

        for (int j=1;j<=m;j++)

            add(st,s[1][i][j],inf),add(s[h+1][i][j],ed,inf);

    for (int k=1;k<=h;k++)

        for (int i=1;i<=n;i++)

            for (int j=1;j<=m;j++)

                scanf("%d",&v),add(s[k][i][j],s[k+1][i][j],v);

    for (int k=1;k<=h;k++)

        for (int i=1;i<=n;i++)

            for (int j=1;j<=m;j++)

                for (int p=0;p<4;p++)

                    if (s[k+D][i+dx[p]][j+dy[p]])

                        add(s[k+D][i+dx[p]][j+dy[p]],s[k][i][j],inf);

    memset(dis,0,sizeof(dis));

    memset(gap,0,sizeof(gap));

    gap[0]=num;

    for (int i=st;i<=ed;i++) last[i]=fir[i];

    int ans=0;

    while (dis[st]<num) ans+=dfs(st,inf);

    printf("%d\n",ans);

    return 0;

}

Manacher

#define N 200005

#include <cstdio>

#include <iostream>

using namespace std;

int cnt, p[N];

char st[N], s[N];

int main(){

    cin>>st;

    s[cnt++]=' ';

    for (int i=0;st[i];i++) s[cnt++]=st[i], s[cnt++]=' ';

    for (int i=0, rad=-1, cen, j;i<cnt;i++){

        if (rad<i) j=0; else j=min(rad-i+1, p[cen\*2-i]);

        for (;i+j<cnt && i>=j && s[i+j]==s[i-j];++j);

        if (i+(p[i]=j)-1>rad)

            rad=i+p[i]-1,

            cen=i;

    }

    int ans=0;

    for (int i=0;i<cnt;i++)

        ans=max(ans,p[i]-1);

    cout<<ans<<endl;

    return 0;

}

SegmentTree\_2D\_单点修改单点查询

int n, m, k, T, x1, yy, x2 ,y2, cnt, x;

//map <int, int> g;

int son[N\*2][5], g[N\*2];

char s[N\*50+5];

int read(){

int p=0;

while (s[x]<'0' || s[x]>'9') x++;

while (s[x]>='0' && s[x]<='9') p=p\*10+s[x++]-'0';

return p;

}

void pushdown(int t){

if (!g[t]) return;

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

if (son[t][i])

if (!g[son[t][i]] || g[son[t][i]]==g[t]) g[son[t][i]]=g[t];

else g[son[t][i]]=-1;

g[t]=0;

}

void upd(int t, int l1, int r1, int l2, int r2){

if (x1<=l1 && r1<=x2 && yy<=l2 && r2<=y2){

if (!g[t] || g[t]==k) g[t]=k;

else g[t]=-1;

return;

}

if (g[t]<0) return;

pushdown(t);

int midx=(l1+r1)>>1, midy=(l2+r2)>>1;

if (x1<=midx && yy<=midy) upd(son[t][0],l1,midx,l2,midy);

if (x1<=midx && y2>midy) upd(son[t][1],l1,midx,midy+1,r2);

if (x2>midx && yy<=midy) upd(son[t][2],midx+1,r1,l2,midy);

if (x2>midx && y2>midy) upd(son[t][3],midx+1,r1,midy+1,r2);

}

int query(int t, int l1, int r1, int l2, int r2){

if (l1==r1 && l2==r2 || g[t]<0) return g[t];

pushdown(t);

int midx=(l1+r1)>>1, midy=(l2+r2)>>1;

if (x1<=midx && yy<=midy) return query(son[t][0],l1,midx,l2,midy);

if (x1<=midx && yy>midy) return query(son[t][1],l1,midx,midy+1,r2);

if (x1>midx && yy<=midy) return query(son[t][2],midx+1,r1,l2,midy);

return query(son[t][3],midx+1,r1,midy+1,r2);

}

void build(int &t, int l1, int r1, int l2, int r2){

if (l1>r1 || l2>r2) return;

t=++cnt;

if (l1==r1 && l2==r2) return;

int midx=l1+r1>>1, midy=l2+r2>>1;

build(son[t][0], l1, midx, l2, midy);

build(son[t][1], l1, midx, midy+1, r2);

build(son[t][2], midx+1, r1, l2, midy);

build(son[t][3], midx+1, r1, midy+1, r2);

}

int main(){

fread(s,1,N\*50,stdin);

n=read();m=read();T=read();

int p, q=n\*m;

build(p,1,n,1,m);

x1=yy=x2=y2=1;

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

k=read(),

upd(1,1,n,1,m);

//cout<<x1<<' '<<yy<<' '<<query(1,1,n,1,m)<<endl;;

if (++yy>m) x1++, yy=1;

if (++y2>m) x2++, y2=1;

}

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

x1=read(),

yy=read(),

x2=read(),

y2=read(),

k=read(),

upd(1,1,n,1,m);

int ans=0;

x1=yy=1;

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

ans+=query(1,1,n,1,m)>=0;

//cout<<x1<<' '<<yy<<' '<<query(1,1,n,1,m)<<endl;

if (++yy>m) x1++, yy=1;

}

cout<<n\*m-ans<<endl;

return 0;

}

Dijkstra+Priority\_Queue

#include <queue>

#define N 100000

#define M 500000

#define INF 999999999

#define num(x) ((x)>='0' && (x)<='9')

typedef unsigned long long ull;

typedef long long ll;

using namespace std;

int n, cnt, m, st, ed, flag[N], nex[M], nu[M], va[M], dist[N];

struct node{

int n, dist;

node (int n, int dist): n(n), dist(dist){}

bool operator <(const node &o) const {return this->dist<o.dist;}

bool operator >(const node &o) const {return this->dist>o.dist;}

};

//priority\_queue<int> qq;//这是个大猪蹄子，大根堆

//typedef pair<int, int> P;

//priority\_queue<P, vector<P>, greater<P> > Q; pair按字典序比较

priority\_queue<node, vector<node>, greater<node> > q;

int read(){

int p=0, q=1;

char ch=getchar();

while (!num(ch)) (ch=='-'?q=-1:0), ch=getchar();

while (num(ch)) p=p\*10+ch-'0', ch=getchar();

return p\*q;

}

void add(int u, int v, int w){

nex[++cnt]=nex[u];nex[u]=cnt;nu[cnt]=v;va[cnt]=w;

}

void dijkstra(){

for (int i=1;i<=n;i++) dist[i]=INF, flag[i]=0;

dist[st]=0;

q.push(node(st,0));

while (!q.empty()){

node curNode=q.top();

q.pop();

int u=curNode.n;

flag[u]=1;

for (int j=nex[u];j;j=nex[j]){

int v=nu[j];

if (!flag[v] && dist[u]+va[j]<dist[v])

dist[v]=dist[u]+va[j],

q.push(node(v,dist[v]));

}

while (!q.empty() && flag[q.top().n]) q.pop();

}

}

int main(){

cnt=n=read();m=read();

st=read();ed=read();

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

int u=read(), v=read(), w=read();

add(u, v, w);

add(v, u, w);

}

dijkstra();

cout<<dist[ed]<<endl;

return 0;

}

DC3\_by\_ez\_zkj

#include <bits/stdc++.h>

#define N 50100

#define F(x) ((x)/3+((x)%3==1?0:tb))

#define G(x) ((x)<tb?(x)\*3+1:((x)-tb)\*3+2)

using namespace std;

char s[N];

int sa[10\*N],rk[N],h[N];

int r[10\*N],wa[10\*N],wb[10\*N],wv[10\*N];

int wws[10\*N];

int n;

void sort(int \*r,int \*a,int \*b,int n,int m)

{

int i;

for(i=0;i<n;i++) wv[i]=r[a[i]];

for(i=0;i<m;i++) wws[i]=0;

for(i=0;i<n;i++) wws[wv[i]]++;

for(i=1;i<m;i++) wws[i]+=wws[i-1];

for(i=n-1;i>=0;i--) b[--wws[wv[i]]]=a[i];

return;

}

int c0(int \*r,int a,int b) {return r[a]==r[b]&&r[a+1]==r[b+1]&&r[a+2]==r[b+2];}

int c12(int k,int \*r,int a,int b)

{

if(k==2) return r[a]<r[b]||(r[a]==r[b]&&c12(1,r,a+1,b+1));

else return r[a]<r[b]||(r[a]==r[b]&&wv[a+1]<wv[b+1]);

}

void dc3(int \*r,int \*sa,int n,int m)

{

int i,j,\*rn=r+n,\*san=sa+n,ta=0,tb=(n+1)/3,tbc=0,p;

r[n]=r[n+1]=0;

for(i=0;i<n;i++) if(i%3!=0) wa[tbc++]=i;

sort(r+2,wa,wb,tbc,m);

sort(r+1,wb,wa,tbc,m);

sort(r,wa,wb,tbc,m);

for(p=1,rn[F(wb[0])]=0,i=1;i<tbc;i++) rn[F(wb[i])]=c0(r,wb[i-1],wb[i])?p-1:p++;

if(p<tbc) dc3(rn,san,tbc,p);

else for(i=0;i<tbc;i++) san[rn[i]]=i;

for(i=0;i<tbc;i++) if(san[i]<tb) wb[ta++]=san[i]\*3;

if(n%3==1) wb[ta++]=n-1;

sort(r,wb,wa,ta,m);

for(i=0;i<tbc;i++) wv[wb[i]=G(san[i])]=i;

for(i=0,j=0,p=0;i<ta && j<tbc;p++)

sa[p]=c12(wb[j]%3,r,wa[i],wb[j])?wa[i++]:wb[j++];

for(;i<ta;p++) sa[p]=wa[i++];

for(;j<tbc;p++) sa[p]=wb[j++];

}

void geth()

{

int j=0,k; h[1]=0;

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

{

k=sa[rk[i]-1];

while (i+j<=n&&k+j<=n&&s[i+j-1]==s[k+j-1]) j++;

h[rk[i]]=j; if (j>0) j--;

}

}

int main()

{

scanf("%s\n",s);

n=strlen(s); int m=255; //s从0开始 n长度 m字符集大小

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

dc3(r,sa,n+1,m+1); //dc3过程后r会被破坏

for (int i=1;i<=n;i++) rk[sa[i]]=i;

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

for (int i=n;i>0;i--) rk[i]=rk[i-1]; //sa、rk均从下标1开始

geth();

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

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

}

虚树+倍增lca+倍增树路径min值

#define LOGN 30

#define N 500050

#define num(x) ((x)>='0' && (x)<='9')

typedef unsigned long long ull;

typedef long long ll;

using namespace std;

const int INF=1999999999;

int \_, n, m, cnt, top, stack[N], flag[N], tag[N], lg[N], h[N];

int nex[N\*5], nu[N\*5], va[N\*5];

int dfn[N], dep[N];

int fa[N][LOGN], g[N][LOGN];

ll f[N];

int read(){

int p=0, q=1;

char ch=getchar();

while (!num(ch)) (ch=='-'?q=-1:0), ch=getchar();

while (num(ch)) p=p\*10+ch-'0', ch=getchar();

return p\*q;

}

void add(int u, int v, int w){

nex[++cnt]=nex[u];nex[u]=cnt;nu[cnt]=v;va[cnt]=w;

}

void dfs(int u, int dad, int w){

dfn[u]=++cnt;

dep[u]=dep[dad]+1;

fa[u][0]=dad;

g[u][0]=w;

for (int j=1;fa[u][j-1];j++)

fa[u][j]=fa[fa[u][j-1]][j-1],

g[u][j]=min(g[u][j-1],g[fa[u][j-1]][j-1]);

for (int j=nex[u];j;j=nex[j]){

int v=nu[j];

if (v==dad) continue;

dfs(v,u,va[j]);

}

}

void initialize(){

lg[1]=0;

for (int i=2;i<=n;i++) lg[i]=lg[i>>1]+1;

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

for (int j=0;j<=lg[n];j++)

g[i][j]=INF;

cnt=0;

dfs(1,0,INF);

}

int get\_lca(int u, int v){

if (dep[u]<dep[v]) swap(u,v);

while (dep[u]>dep[v]) u=fa[u][lg[dep[u]-dep[v]]];

if (u==v) return u;

for (int j=lg[n];j>=0;j--)

if (fa[u][j]==fa[v][j]) continue;

else u=fa[u][j], v=fa[v][j];

return fa[u][0];

}

int get\_min(int u, int v){

//cout<<u<<' '<<v<<' ';

int mi=INF;

while (u!=v){

mi=min(mi,g[v][lg[dep[v]-dep[u]]]);

v=fa[v][lg[dep[v]-dep[u]]];

}

//cout<<mi<<endl;

return mi;

}

void link(int u, int v){

if (tag[u]!=\_) tag[u]=\_, nex[u]=0;

if (tag[v]!=\_) tag[v]=\_, nex[v]=0;

add(u,v,get\_min(u,v));

}

void pop(){

int v=stack[top--];

//cout<<v<<endl;

if (!top) return;

int u=stack[top];

link(u,v);

}

void push(int u){

stack[++top]=u;

}

void build\_vt(){

cnt=n;

stack[top=1]=1;

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

int u=h[i], v=stack[top];

int LCA=get\_lca(u,v);

//cout<<u<<' '<<v<<' '<<LCA<<' '<<endl;

while (top>1 && dep[stack[top-1]]>=dep[LCA]) pop();

if (stack[top]!=LCA){

link(LCA,stack[top]);

top--;

push(LCA);

}

push(u);

}

while (top) pop();

}

bool cmp(int a, int b) { return dfn[a]<dfn[b];}

void dp(int u){

f[u]=0;

for (int j=nex[u];j;j=nex[j]){

int v=nu[j];

if (flag[v]==\_) f[v]=INF;else dp(v);

f[u]+=min((ll)va[j],f[v]);

}

//cout<<u<<' '<<f[u]<<endl;

}

void solve(){

m=read();

for (int i=1;i<=m;i++) h[i]=read(), flag[h[i]]=\_;

sort(h+1,h+1+m,cmp);

build\_vt();

dp(1);

printf("%lld\n", f[1]);

}

int main(){

cnt=n=read();

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

int u=read(), v=read(), w=read();

add(u,v,w);

add(v,u,w);

}

initialize();

for (\_=read();\_;\_--) solve();

return 0;

}

可持久化并查集+启发式合并: O(nlog^2n)

#include<bits/stdc++.h>

#define max(x,y) ((x)>(y)?(x):(y))

#define min(x,y) ((x)<(y)?(x):(y))

#define LL long long

#define swap(x,y) (x^y?(x^=y,y^=x,x^=y):0)

#define tc() (A==B&&(B=(A=ff)+fread(ff,1,100000,stdin),A==B)?EOF:\*A++)

#define pc(ch) (pp\_<100000?pp[pp\_++]=(ch):(fwrite(pp,1,100000,stdout),pp[(pp\_=0)++]=(ch)))

#define N 200000

int pp\_=0;char ff[100000],\*A=ff,\*B=ff,pp[100000];

using namespace std;

int n,Q,tot=0,rt[N+5],a[N+5];

struct Chairman\_Tree

{

int Son[2],fa,level;

}node[N\*20];

inline void read(int &x)

{

x=0;int f=1;char ch;

while(!isdigit(ch=tc())) f=ch^'-'?1:-1;

while(x=(x<<3)+(x<<1)+ch-'0',isdigit(ch=tc()));

x\*=f;

}

inline void write(int x)

{

if(x<0) pc('-'),x=-x;

if(x>9) write(x/10);

pc(x%10+'0');

}

inline void Build(int &rt,int l,int r)//初始的建树，一开始每个节点的fa都是本身，这是并查集的基础思想

{

rt=++tot;

int mid=l+r>>1;

if(!(l^r)) {node[rt].fa=l;return;}

Build(node[rt].Son[0],l,mid),Build(node[rt].Son[1],mid+1,r);

}

inline void NewPoint(int &rt,int lst,int l,int r,int x,int fa)//新插入一个节点

{

rt=++tot;

int mid=l+r>>1;

if(!(l^r)) {node[rt].fa=fa,node[rt].level=node[lst].level;return;}//更新fa，并复制以前版本的这个节点的level

node[rt].Son[0]=node[lst].Son[0],node[rt].Son[1]=node[lst].Son[1];

if(x<=mid) NewPoint(node[rt].Son[0],node[lst].Son[0],l,mid,x,fa);

else NewPoint(node[rt].Son[1],node[lst].Son[1],mid+1,r,x,fa);

}

inline void Add\_level(int rt,int l,int r,int x)//增加一个节点的在按秩合并时的优先级

{

int mid=l+r>>1;

if(!(l^r)) {++node[rt].level;return;}

if(x<=mid) Add\_level(node[rt].Son[0],l,mid,x);

else Add\_level(node[rt].Son[1],mid+1,r,x);

}

inline int Query(int rt,int l,int r,int x)//询问x节点在某一版本下的位置

{

int mid=l+r>>1;

if(!(l^r)) return rt;

if(x<=mid) return Query(node[rt].Son[0],l,mid,x);

else return Query(node[rt].Son[1],mid+1,r,x);

}

inline int getfa(int rt,int x)//询问x节点在某一版本下的祖先

{

int fa=Query(rt,1,n,x);

return node[fa].fa^x?getfa(rt,node[fa].fa):fa;//如果x节点在该版本下的父亲等于它本身，就返回x，否则返回x的父亲在这个版本下的祖先，和经典的getfa()函数差不多

}

inline void connect(int v,int x,int y)//在版本v中连接x和y，将他们放入一个集合中

{

int fx=getfa(rt[v],x),fy=getfa(rt[v],y);//先求出版本v中它们的祖先

if(!(fx^fy)) return;//如果祖先相同，就退出函数

if(node[fx].level<node[fy].level) swap(fx,fy);//如果x的优先级小于y的优先级，就交换x和y

NewPoint(rt[v],rt[v-1],1,n,node[fy].fa,node[fx].fa);//将优先级小的节点的父亲连向优先级大的节点的父亲

if(!(node[fx].level^node[fy].level)) Add\_level(rt[v],1,n,node[fx].fa);//如果它们的优先级相同，就将它们合并后的祖宗的优先级加1

}

int main()

{

register int i;

for(read(n),read(Q),Build(rt[0],i=1,n);i<=Q;++i)//先建一棵树，然后进行操作

{

int op,x,y;read(op),read(x);

if(op^2) read(y),rt[i]=rt[i-1];

switch(op)

{

case 1:connect(i,x,y);break;//在当前版本下连接x和y

case 2:rt[i]=rt[x];break;//将当前版本还原回曾经的版本x

case 3:pc(getfa(rt[i],x)^getfa(rt[i],y)?'0':'1'),pc('\n');break;//若当前版本下x和y的父亲相同，输出1，否则输出0

}

}

return fwrite(pp,1,pp\_,stdout),0;

}

# 可持久化并查集

#define inf 2000000100

#define N 400005

#define M N\*3

#define NLGN N\*31

int n, m, cnt, root[N];

int size[NLGN], fa[NLGN], ls[NLGN], rs[NLGN], tag[NLGN], g[NLGN];

struct edge{

    int u, v, w, a;

    bool operator <(const edge &o) const {return this->a>o.a;}

    bool operator >(const edge &o) const {return this->a<o.a;}

}l[N];

int nex[M], nu[M], va[M];

struct node{

    int n, dist;

    node (int n, int dist): n(n), dist(dist){}

    bool operator <(const node &o) const {return this->dist<o.dist;}

    bool operator >(const node &o) const {return this->dist>o.dist;}

};

int read(){

    int p=0, q=1;

    char ch=getchar();

    while (ch<'0' || ch>'9') ch=='-'?q=-1:0, ch=getchar();

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

    return p\*q;

}

void link(int u, int v, int w){

    nex[++cnt]=nex[u];nex[u]=cnt;nu[cnt]=v;va[cnt]=w;

}

int flag[N], dist[N];

priority\_queue<node, vector<node>, greater<node> > q;

void dijkstra(int st){

    for (int i=1;i<=n;i++) dist[i]=inf, flag[i]=0;

    dist[st]=0;

    q.push(node(st,0));

    while (!q.empty()){

        node curNode=q.top();

        q.pop();

        int u=curNode.n;

        flag[u]=1;

        for (int j=nex[u];j;j=nex[j]){

            int v=nu[j];

            if (!flag[v] && dist[u]+va[j]<dist[v])

                dist[v]=dist[u]+va[j],

                q.push(node(v,dist[v]));

        }

        while (!q.empty() && flag[q.top().n]) q.pop();

    }

}

int query(int t, int l, int r, int x){

    if (l==r) return t;

    int mid=l+r>>1;

    if (x<=mid) return query(ls[t],l,mid,x);

    return query(rs[t],mid+1,r,x);

}

int find(int t, int x){ //t is root\_address

    int v=query(t,1,n,x);

    if (x==fa[v]) return v;

    return find(t,fa[v]);

}

int new\_node(){ //multi\_test

    ++cnt;

    fa[cnt]=size[cnt]=g[cnt]=ls[cnt]=rs[cnt]=tag[cnt]=0;

    return cnt;

}

void build(int &t, int l, int r){

    t=new\_node();

    if (l==r){

        fa[t]=l;

        size[t]=1;

        g[t]=dist[l];

        return;

    }

    int mid=l+r>>1;

    build(ls[t],l,mid);

    build(rs[t],mid+1,r);

}

void insert(int u, int v, int x, int y, int z, int dis){

    int tv=v;

    int le=1, ri=n;

    while (le<ri){

        int mid=le+ri>>1;

        if (x<=mid){

            if (!rs[v]) rs[v]=rs[u];

            if (!ls[v] || tag[ls[v]]!=tv) ls[v]=new\_node(), tag[cnt]=tv;

            v=ls[v],

            u=ls[u],

            ri=mid;

        }

        else{

            if (!ls[v]) ls[v]=ls[u];

            if (!rs[v] || tag[rs[v]]!=tv) rs[v]=new\_node(), tag[cnt]=tv;

            v=rs[v],

            u=rs[u],

            le=mid+1;

        }

    }

    fa[v]=y;

    size[v]=z;

    g[v]=dis;

}

int get(int p){

    int le=0, ri=m+1;

    while (le<ri-1){

        int mid=le+ri>>1;

        if (l[mid].a>p) le=mid;

        else ri=mid;

    }

    return le;

}

void solve(){

    cnt=n=read();m=read();

    for (int i=1;i<=n;i++) nex[i]=0;

    for (int i=1;i<=m;i++){

        int u=read(), v=read(), w=read(), a=read();

        l[i]=edge{u,v,w,a};

        link(u,v,w);

        link(v,u,w);

    }

    dijkstra(1);

    cnt=0;

    build(root[0],1,n);

    sort(l+1,l+1+m);

    for (int i=1;i<=m;i++){

        int u=l[i].u, v=l[i].v;

        u=find(root[i-1],u);

        v=find(root[i-1],v);

        root[i]=root[i-1];

        if (fa[u]==fa[v]) continue;

        root[i]=new\_node();

        if (size[u]>size[v]) swap(u,v);

        insert(root[i-1], root[i], fa[u], fa[v], size[u], g[u]);

        insert(root[i-1], root[i], fa[v], fa[v], size[u]+size[v], min(g[u], g[v]));

    }

    int ans=0;

    int Q=read(), K=read(), S=read();

    for (int i=1;i<=Q;i++){

        int v=(read()+1ll\*K\*ans-1)%n+1;

        int p=(read()+1ll\*K\*ans)%(S+1);

        int rt=get(p);

        int par=find(root[rt],v);

        printf("%d\n", ans=g[find(root[get(p)],v)]);

    }

}

int main(){

    for (int \_=read();\_;\_--) solve();

    return 0;

}

Dsu\_on\_tree

#define N 100005

#define M N \* 3

ll sum, ans[N];

int n, cnt, Son, ma;

int g[N], c[N], size[N], son[N];

int nex[M], nu[M];

void link(int u, int v)

{

    nex[++cnt] = nex[u];

    nex[u] = cnt;

    nu[cnt] = v;

}

void dfs\_cut(int u, int fa)

{

    size[u] = 1;

    son[u] = 0;

    int mx = 0;

    for (int j = nex[u]; j; j = nex[j])

    {

        int v = nu[j];

        if (v == fa)

            continue;

        dfs\_cut(v, u);

        if (size[v] > mx)

            mx = size[v], son[u] = v;

        size[u] += size[v];

    }

}

void calc(int u, int fa, int val)

{

    g[c[u]] += val;

    if (g[c[u]] > ma)

        ma = g[c[u]], sum = c[u];

    else if (g[c[u]] == ma)

        sum += c[u];

    for (int j = nex[u]; j; j = nex[j])

    {

        int v = nu[j];

        if (v == fa || v == Son)

            continue;

        calc(v, u, val);

    }

}

void dsu\_ot(int u, int fa, int opt)

{

    for (int j = nex[u]; j; j = nex[j])

    {

        int v = nu[j];

        if (v == fa || v == son[u])

            continue;

        dsu\_ot(v, u, 0);

    }

    if (son[u])

        dsu\_ot(son[u], u, 1), Son = son[u];

    calc(u, fa, 1);

    Son = 0;

    ans[u] = sum;

    if (!opt)

        calc(u, fa, -1), sum = 0, ma = 0;

}

int main()

{

    cnt = n = read();

    for (int i = 1; i <= n; i++)

        c[i] = read();

    for (int i = 1; i < n; i++)

    {

        int u = read(), v = read();

        link(u, v);

        link(v, u);

    }

    dfs\_cut(1, 0);

    dsu\_ot(1, 0, 0);

    for (int i = 1; i <= n; i++)

        printf("%I64d%s", ans[i], i == n ? "\n" : " ");

    return 0;

}

# \_\_int128: (need linux)

#include <bits/stdc++.h>

using namespace std;

//\_\_int128: -2^126~2^126

inline \_\_int128 read()

{

\_\_int128 x=0,f=1;

char ch=getchar();

while(ch<'0'||ch>'9')

{

if(ch=='-')

f=-1;

ch=getchar();

}

while(ch>='0'&&ch<='9')

{

x=x\*10+ch-'0';

ch=getchar();

}

return x\*f;

}

inline void write(\_\_int128 x)

{

if(x<0)

{

putchar('-');

x=-x;

}

if(x>9)

write(x/10);

putchar(x%10+'0');

}

int main()

{

\_\_int128 a = read();

\_\_int128 b = read();

write(a + b);

return 0;

}

# k**-d tree**

#define K 2

#define N 1000010

#define inf 100000000

#define DATA N\*22

typedef long long ll;

int pos;

char s[DATA+1];

int D;

struct Tree\_Point{

    int tag, D;

    int d[K];

    int ls, rs;

    int min\_d[K], max\_d[K];

}t[N];

struct Point{   //include operator && sort\_point

    int d[K];

    int tag, op;

    bool operator <(const Point &o) const {

        return d[D]==o.d[D]?tag<o.tag:d[D]<o.d[D];

    }

    bool operator <(const Tree\_Point &o) const {

        return d[D]==o.d[D]?tag<o.tag:d[D]<o.d[D];

    }

}a[N], op[N/2];

void update(int x){

    int ls=t[x].ls, rs=t[x].rs;

    for (int j=0;j<K;j++)

        t[x].min\_d[j]=std::min(t[x].min\_d[j], std::min(t[ls].min\_d[j],t[rs].min\_d[j])),

        t[x].max\_d[j]=std::max(t[x].max\_d[j], std::max(t[ls].max\_d[j],t[rs].max\_d[j]));

}

int build(int l, int r, int d){

    D=d;if (D==K) D=d=0;

//  D=rand()%K;

//  std::cout<<"Build: "<<l<<' '<<r<<' '<<d<<std::endl;

    int mid=l+r>>1;

    std::nth\_element(a+l,a+mid,a+r+1);

    t[mid].tag=a[mid].tag;

    t[mid].D=D;

    for (int j=0;j<K;j++){

        t[mid].d[j]=a[mid].d[j];

        if (!t[mid].tag)

            t[mid].min\_d[j]=t[mid].max\_d[j]=a[mid].d[j];

        else

            t[mid].min\_d[j]=-(t[mid].max\_d[j]=-inf);

    }

    if (l<mid) t[mid].ls=build(l,mid-1,d+1);

    if (r>mid) t[mid].rs=build(mid+1,r,d+1);

    update(mid);

    return mid;

}

int tag;

void activate(int x){

    D=t[x].D;

//  std::cout<<"Activating: "<<x<<' '<<d<<std::endl;

    if (tag==t[x].tag){

        for (int j=0;j<K;j++)

            t[x].min\_d[j]=std::min(t[x].min\_d[j], t[x].d[j]),

            t[x].max\_d[j]=std::max(t[x].max\_d[j], t[x].d[j]);

        return;

    }

    if (op[tag]<t[x])

        activate(t[x].ls);

    else activate(t[x].rs);

    update(x);

}

int getdist(int x){

    int res=0;

    for (int j=0;j<K;j++)

        res+=std::max(t[x].min\_d[j]-op[tag].d[j],0)

            +std::max(op[tag].d[j]-t[x].max\_d[j],0);

    return res;

}

int ans;

int query(int x){

    int tmp=0, dls, drs;

    int ls=t[x].ls, rs=t[x].rs;

    if (t[x].tag<=tag)

        for (int j=0;j<K;j++) tmp+=abs(t[x].d[j]-op[tag].d[j]);

    else tmp=inf;

    if (ls) dls=getdist(ls); else dls=inf;

    if (rs) drs=getdist(rs); else drs=inf;

    if (tmp<ans) ans=tmp;

    tmp=dls<drs;

    if (tmp){

        if (dls<ans) query(ls);

        if (drs<ans) query(rs);

    }

    else{

        if (drs<ans) query(rs);

        if (dls<ans) query(ls);

    }

}

int main(){

    fread(s,1,DATA,stdin);

    srand(unsigned(time(NULL)));

    int n=read(), m=read();

    for (int i=1;i<=n;i++)

        for (int j=0;j<K;j++) a[i].d[j]=read();

    for (int i=1;i<=m;i++){

        op[i].tag=i;

        op[i].op=read();

        for (int j=0;j<K;j++) op[i].d[j]=read();

        if (op[i].op==1) a[++n]=op[i];

    }

    for (int j=0;j<K;j++)

        t[0].min\_d[j]=-(t[0].max\_d[j]=-inf);

    int root=build(1,n,0);

//  std::cout<<"Operating!"<<' '<<n<<' '<<m<<std::endl;

    for (tag=1;tag<=m;tag++)

        if (op[tag].op==1)

            activate(root);

        else{

            ans=inf-1;

            query(root);

            printf("%d\n", ans);

        }

    return 0;

}

# 极角排序

struct node{

    int x, y, g, xx;

    long double thi, cs;

}l[N];

inline ll xj(int i, int j){

    return 1ll\*l[i].x\*l[j].y-1ll\*l[i].y\*l[j].x;

}

inline ll cross\_dot(int x1, int y1, int x2, int y2){

    return 1ll\*x1\*y2-1ll\*x2\*y1;

}

inline bool cmp(node &a, node &b){

//  method1 (bad eps)

    return a.thi<b.thi;

//  method2 (no eps)

    if (a.xx<b.xx) return 1;

    if (a.xx>b.xx) return 0;

    return cross\_dot(a.x, a.y, b.x, b.y)>0;

// method3 (idk, same bad eps)

    if (a.xx<b.xx) return 1;

    if (a.xx>b.xx) return 0;

    if (a.xx<=2) return a.cs>b.cs;

    return a.cs<b.cs;

}

int get\_xx(int x, int y){

    if (y>=0 && x>0) return 1;

    if (x<=0 && y>0) return 2;

    if (y<=0 && x<0) return 3;

    if (x>=0 && y<0) return 4;

}

inline ll sqr(int a) {return 1ll\*a\*a;}

void solve(int u){

    ct=0;

    for (int i=1;i<=n;i++)

        if (i!=u){

            l[++ct].g=g[i];

            l[ct].x=x[i]-x[u];

            l[ct].y=y[i]-y[u];

            l[ct].thi=atan2l(l[ct].y, l[ct].x);

            l[ct].cs=(long double) l[ct].x/sqrt(sqr(l[ct].x)+sqr(l[ct].y));

            l[ct].xx=get\_xx(l[ct].x, l[ct].y);

        }

    sort(l+1,l+1+ct,cmp);

}

# ExKmp

const int maxn=10086; //字符串长度最大值

int next[maxn],ex[maxn]; //ex数组即为extend数组

void GETNEXT(char \*str){

int i=0,j,po,len=strlen(str);

next[0]=len;//初始化next[0]

while(str[i]==str[i+1]&&i+1<len)//计算next[1]

i++;

next[1]=i;

po=1;//初始化po的位置

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

if(next[i-po]+i<next[po]+po)//第一种情况，可以直接得到next[i]的值

next[i]=next[i-po];

else{//第二种情况，要继续匹配才能得到next[i]的值

j=next[po]+po-i;

if(j<0)j=0;//如果i>po+next[po],则要从头开始匹配

while(i+j<len&&str[j]==str[j+i])//计算next[i]

j++;

next[i]=j;

po=i;//更新po的位置

}

}

}

void EXKMP(char \*s1,char \*s2){

int i=0,j,po,len=strlen(s1),l2=strlen(s2);

GETNEXT(s2);//计算子串的next数组

while(s1[i]==s2[i]&&i<l2&&i<len)//计算ex[0]

i++;

ex[0]=i;

po=0;//初始化po的位置

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

if(next[i-po]+i<ex[po]+po)//第一种情况，直接可以得到ex[i]的值

ex[i]=next[i-po];

else{//第二种情况，要继续匹配才能得到ex[i]的值

j=ex[po]+po-i;

if(j<0)j=0;//如果i>ex[po]+po则要从头开始匹配

while(i+j<len&&j<l2&&s1[j+i]==s2[j])//计算ex[i]

j++;

ex[i]=j;

po=i;//更新po的位置

}

}

}

# Compare\_In\_Linux

#!/bin/sh

echo '1' > p1537.out

echo '1' > std.out

g++ data.cpp -o data

g++ std.cpp -o std

g++ p1537.cpp -o p1537

while (diff p1537.out std.out) do

echo '==data=='

./data

echo '==std=='

./std

echo '==p1537=='

./p1537

done;

# vimrc

set nu

set cindent

set tabstop=4

set shiftwidth=4

set mouse=a

# vimrc2

syntax on

set nu ru ar ic si sta et

set mouse=a

set hi=1000

set sw=4

set ts=4

set sts=4

nmap ; :

imap {<CR> {<ESC>o}<ESC>O

nmap <F4> :w<CR> :!g++ -DLOCAL -Wall -Wextra -pedantic -std=c++11 -o a.ao % -O2 <CR>

nmap <F5> :w<CR> :!g++ -DLOCAL -Wall -Wextra -pedantic -std=c++11 -o a.ao % -O2 && ./a.ao <CR>

nmap <F7> :w<CR> :!g++ -DLOCAL -Wall -Wextra -pedantic -std=c++11 -o a.ao % -g && gdb ./a.ao <CR>

nmap <F8> :w<CR> :!g++ -DLOCAL -Wall -Wextra -pedantic -std=c++11 -o a.ao % -O2 -Wfatal-errors && ./a.ao <CR>

nmap <F9> :w<CR> :!g++ -DLOCAL -Wall -Wextra -pedantic -std=c++11 -o a.ao % -O2 -fsanitize=address && ./a.ao <CR>

set guifont=Monospace\ 14

vmap <C-X> x

vmap <C-C> y

map <C-V> gP

imap <C-V> <C-O>gP

nmap <C-S> :update<CR>

imap <C-S> <Esc>:update<CR>gi

nmap <C-Z> u

imap <C-Z> <C-O>u

nmap <C-Y> <C-R>

imap <C-Y> <C-O><C-R>

nmap <C-A> gggH<C-O>G

imap <C-A> <C-O>gg<C-O>gH<C-O>G

nmap <C-D> dd

imap <C-D> <C-O>dd