IO

```
namespace IO {
    const unsigned int bufsize=1<<16,outsize=1<<20;
    static char ch[bufsize],*S=ch,*T=ch;
    inline char getc()
    {return ((S==T)&&(T=(S=ch)+fread(ch,1,bufsize,stdin),S==T)?0:*S++);}
    static char Out[outsize],*nowp=Out;
    inline void flush(){fwrite(Out,1,nowp-Out,stdout);nowp=Out;}
    template<typename T> void read(T &x) {
            char c=getc();x=0;
            for(;!isdigit(c);c=getc());
            for(;isdigit(c);x=(x<<1)+(x<<3)+(c^'0'),c=getc());
        }
    void readstr(char *s) {
        char c=getc();static int slen=0;
        f(i,1,slen)s[i]=0;slen=0;
        for(;!isalpha(c);c=getc());
        for(;isalpha(c);c=getc())s[++slen]=c;
    }
    template<typename T> void write(T x,char c='\n') {
            if(!x)*nowp++='0';
            if(x<0)*nowp++='-',x=-x;
            static unsigned int stk[50],tp=0;
            for(;x;x/=10)stk[++tp]=x%10;
            for(;tp;*nowp++=stk[tp--]^'0');*nowp++=c;
        }
}
```

KMP

```
void getborder(char *s) {
    int n=strlen(s+1);
    nex[0]=-1;
    f(i,2,n) {
        int x=nex[i-1];
        while(x!=-1)
```

```
        {if(s[x+1]==s[i]){nex[i]=x+1;break;}x=nex[x];}
    }
}
char s[1000010],t[1000010];
int main() {
    getborder(t);
    int x=0;
    f(i,1,n) {
        while(x&&t[x+1]^s[i])x=nex[x];
        ++x;
        if(x==m)printf("%d\n",i-m+1);
    }
    f(i,1,m)printf("%d%c",nex[i],i^iend?' ':'\n');
}
```

LCT

```
typedef ui arr[neko];
arr Mul,Add,Sum,Val,Siz,Rev;
ui ADD(ui x,ui y){return (x+=y)>=mod?x-=mod:x;}
int n,m,son[neko][2],fa[neko],s[neko];
namespace LCT {
    int tp;
    void pushr(int x)
    {std::swap(son[x][0],son[x][1]),Rev[x]^=1;}
    void pusha(int x,ui v)
    {
        Val[x]=ADD(Val[x],v);
        Sum[x]=ADD(Sum[x],v*Siz[x]%mod);
        Add[x]=ADD(Add[x],v);
    }
    void pushm(int x,ui v)
    {
        Val[x]=Val[x]*v%mod;
        Sum[x]=Sum[x]*v%mod;
        Add[x]=Add[x]*v%mod;
        Mul[x]=Mul[x]*v%mod;
        //printf("%u %u %u\n",Sum[x],Add[x],Mul[x]);
    }
    void pushup(int x)
    {
        Sum[x]=ADD(Val[x],ADD(Sum[son[x][0]],Sum[son[x][1]]));
        Siz[x]=Siz[son[x][0]]+Siz[son[x][1]]+1;
    }
    void pushdown(int x)
    {
        if(Rev[x])pushr(son[x][0]),pushr(son[x][1]),Rev[x]=0;
        if(Mul[x]^1)pushm(son[x][0],Mul[x]),pushm(son[x][1],Mul[x]),Mul[x]=1;
        if(Add[x])pusha(son[x][0],Add[x]),pusha(son[x][1],Add[x]),Add[x]=0;
    }
    bool isroot(int x)
    {return (son[fa[x]][0]^x)&&(son[fa[x]][1]^x);}
    bool get(int x)
    {return son[fa[x]][1]==x;}
    void rotate(int x)
    {
        int y=fa[x],z=fa[y];bool side=get(x);
```

```
        if(!isroot(y))son[z][get(y)]=x;fa[x]=z;
        fa[son[y][side]=son[x][side^1]]=y;
        fa[son[x][side^1]=y]=x;
        pushup(y),pushup(x);
    }
    void splay(int x)
    {
        int y;
        s[tp=1]=x;
        for(y=x;!isroot(y);y=fa[y])s[++tp]=fa[y];
        while(tp)pushdown(s[tp--]);

for(y=fa[x];!isroot(x);rotate(x),y=fa[x])if(!isroot(y))rotate(get(x)^get(y)?x:y);
        pushup(x);
    }
    void access(int x)
    {for(register int y=0;x;y=x,x=fa[x])splay(x),son[x][1]=y,pushup(x);}
    void makeroot(int x)
    {access(x),splay(x),pushr(x);}
    void split(int x,int y)
    {makeroot(x),access(y),splay(y);}
    void link(int x,int y)
    {makeroot(x),fa[x]=y;}//legal defaultly
    void cut(int x,int y)
    {split(x,y),fa[x]=son[y][0]=0,pushup(y);}
}
int main() {
    using namespace LCT;
    char opt[10];
    int x,y,o;ui z;
    scanf("%d%d",&n,&m);
    f(i,1,n)Val[i]=Mul[i]=1;
    f(i,2,n)scanf("%d%d",&x,&y),link(x,y);
    f(i,1,m)
    {
        scanf("%s%d%d",opt,&x,&y);
        if(opt[0]=='+')scanf("%u",&z),split(x,y),pusha(y,z);
        if(opt[0]=='-')scanf("%u%d",&z,&o),cut(x,y),link(z,o);
        if(opt[0]=='*')scanf("%u",&z),split(x,y),pushm(y,z);
        if(opt[0]=='/')split(x,y),printf("%u\n",ADD(Sum[y],mod));
    }return 0;
}
```

李超树

```
//CF932F.部分内容详见博客.
#define travel(i,u,v) for(register int i=head[u],v=e[i].v;i;i=e[i].nex,v=e[i].v)
const int neko=100010,steko=neko*17*2;
typedef long long ll;
int n,m,t,head[neko];
ll a[neko],b[neko],dp[neko];
int L[steko],R[steko],rt[steko];
struct Line
{ll k,b;int id;}Adv[steko];
struct node
{int v,nex;}e[neko<<1];
void add(int x,int y)
```

```
{
    e[++t].v=y,e[t].nex=head[x],head[x]=t;
    e[++t].v=x,e[t].nex=head[y],head[y]=t;
}
namespace LiC_Tree
{
#define lson L[root],l,mid
#define rson R[root],mid+1,r
    const int lim=200000,las=100000;
    int cnt=0;
    using std::swap;
    ll F(Line y,int x)
    {return y.k*x+y.b;}
    bool check(Line A,Line B,int x)
    {return F(A,x)<=F(B,x);}
    void cover(int &root,int l,int r,Line y)
    {
        if(!root)root=++cnt;
        int mid=l+r>>1;
        //printf("%d %lld %lld %lld
%lld\n",mid,Adv[root].k,y.k,F(Adv[root],mid),F(y,mid));
        if(!Adv[root].id||check(y,Adv[root],mid))swap(Adv[root],y);
        if(!y.id||l==r||Adv[root].k==y.k)return;
        if(Adv[root].k<y.k)cover(lson,y);
        else cover(rson,y);
    }
    Line query(int root,int l,int r,int tag)
    {
        if(l==r)return Adv[root];
        int mid=l+r>>1;Line tmp;
        if(tag<=mid)tmp=query(lson,tag);
        else tmp=query(rson,tag);
        if(tmp.id)return check(tmp,Adv[root],tag)?tmp:Adv[root];
        else return Adv[root];
    }
    int merge(int x,int y,int l,int r)
    {
        if(!x||!y)return x|y;
        int mid=l+r>>1;
        L[x]=merge(L[x],L[y],l,mid);
        R[x]=merge(R[x],R[y],mid+1,r);
        //printf("merging %d\n",Adv[y].id);
        cover(x,l,r,Adv[y]);
        return x;
    }
    void dfs(int u,int fa)
    {
        travel(i,u,v)if(v^fa)dfs(v,u),rt[u]=merge(rt[u],rt[v],1,lim);
        int x=query(rt[u],1,lim,a[u]+las).id;//x=u后代而不是儿子
        dp[u]=a[u]*b[x]+dp[x];
        cover(rt[u],1,lim,(Line){b[u],dp[u]-b[u]*las,u});//k(x-las)+b
        //printf("%d %d\n",u,query(rt[u],1,lim,a[1]+las).id);
    }
}
int main()
{
    int x,y;
    scanf("%d",&n);
```

```
    f(i,1,n)scanf("%lld",&a[i]);
    f(i,1,n)scanf("%lld",&b[i]);
    f(i,2,n)scanf("%d%d",&x,&y),add(x,y);
    memset(Adv,0,sizeof Adv);
    LiC_Tree::dfs(1,0);
    f(u,1,n)printf("%lld%c",dp[u],u^uend?' ':'\n');
}
```

点分树

```
//该题为2018-08-07 NOIp Simulation T3: revive by dy0607.具体内容见pdf。
#include<cstdio>
#include<iostream>
#define neko 300010
#define cmax(a,b) ((a)>(b)?(a):(b))
#define f(i,a,b) for(register int i=(a);i<=(b);i=-(~(i)))
#define travel(i,u,v) for(register int i=head[u],v=e[i].v;i;i=e[i].nex,v=e[i].v)
typedef unsigned long long ull;
namespace IO
{
    const unsigned int bufsize=1<<16,outsize=1<<24;
    static char ch[bufsize],*S=ch,*T=ch;
    inline char getc()
    {return ((S==T)&&(T=(S=ch)+fread(ch,1,bufsize,stdin),S==T)?0:*S++);}
    static char Out[outsize],*nowp=Out;
    inline void flush(){fwrite(Out,1,nowp-Out,stdout);nowp=Out;}
    template<typename T>
    void read(T &x)
    {
        char c=getc();x=0;
        for(;!isdigit(c);c=getc());
        for(;isdigit(c);x=(x<<1)+(x<<3)+(c^'0'),c=getc());
    }
    template<typename T>
    void write(T x,char c='\n')
    {
        if(!x)*nowp++='0';
        static unsigned int stk[50],tp=0;
        for(;x;x/=10)stk[++tp]=x%10;
        for(;tp;*nowp++=stk[tp--]^'0');*nowp++=c;
    }
}
using namespace IO;
struct node
{int v,nex;}e[neko];
int n,q,t=3,anc[neko][20],bln[neko][20],sum[neko][20],coef[neko][20];//bln:这条边在当前点分树深度下属于哪条边,anc是点;sum:这个分治重心当前子树的答案和(sub_i*val_i),方便计算答案;t=3是因为每条边编号/2正好等于存了的权值编号
typedef int arr[neko];
arr siz,msz,sub,psb,cut,ans,val,dep,head;//sub:这条边这个方向下的siz,coef代表当前点分树结构下的这条边的sub;psb:经过当前点的路径条数(passby);cut:是否处于当前点分树内;ans:这个分治重心的答案和(sigma(sum));
//这个ans*sum算贡献有点类似铁人两项，把两两子树的贡献累起来
ull nowans;
void add(int x,int y)
```

```
{
    e[++t].v=y,e[t].nex=head[x],head[x]=t;
    e[++t].v=x,e[t].nex=head[y],head[y]=t;
}
namespace NDC_Tree
{
    #define travel(i,u,v) for(register int
i=head[u],v=e[i].v;i;i=e[i].nex,v=e[i].v)
    int root=0,mod=1e9+7,nown;
    template<typename T,typename L>T ADD(T a,L b){return ((a+=b)>=mod)?a-mod:a;}
    void dfsinit(int u,int fa)
    {
        siz[u]=1;
        travel(i,u,v)
        {
            if(v^fa)
            {
                dfsinit(v,u);
                siz[u]+=siz[v];
                psb[v]=1ll*(sub[i]=siz[v])*(sub[i^1]=(n-siz[v]))%mod;
                nowans=ADD(nowans,1ll*psb[v]*val[v]%mod*val[v]%mod);//val[v]指的是
v代表的这条边的权值
            }
        }
    }
    void dfsroot(int u,int fa)
    {
        siz[u]=1,msz[u]=0;

travel(i,u,v)if(!cut[v]&&v^fa)dfsroot(v,u),siz[u]+=siz[v],msz[u]=cmax(msz[u],siz[
v]);
        msz[u]=cmax(msz[u],nown-siz[u]);
        if(msz[u]<msz[root])root=u;
    }
    void dfscalc(int u,int edge,int depth,int from)
    {
        register int w=edge>>1;
        anc[w][dep[w]=depth]=root,bln[w][depth]=from;
        sum[from][depth]=ADD(sum[from][depth],1ll*(coef[w]
[depth]=sub[edge])*val[w]%mod);//当前重心边管辖的子树各边的答案和累到sum上
        if(cut[u])return;
        travel(i,u,v)if(i^(edge^1))dfscalc(v,i,depth,from);//同v^fa
    }
    void dfs(int u,int depth)//depth:点分树上深度;
    {
        cut[u]=1;
        travel(i,u,v)
        {
            register int from=i>>1;
            dfscalc(v,i,depth,from);
            nowans=ADD(nowans,2ll*sum[from][depth]*ans[u]%mod);//把过重心的所有链答
案算起来
            ans[u]=ADD(ans[u],sum[from][depth]);//累上这个点的答案
        }

travel(i,u,v)if(!cut[v])root=0,nown=siz[v],dfsroot(v,u),dfs(root,depth+1);
    }
    void update(int edge,int addx)//edge代表(fa[edge],edge)这条边，注意不要混淆成点
```

```
    {
        register int u,from,delta=0;//from:edge这条边在当前分治结构下属于的边
        f(i,0,dep[edge])//遍历点分树
        {
            u=anc[edge][i],from=bln[edge][i];
            if(!i||(coef[edge][i]^coef[edge][i-1]))delta=1ll*coef[edge]
[i]*addx%mod;//算delta,相同没必要重复算
            nowans=ADD(nowans,2ll*(ans[u]-sum[from][i]+mod)*delta%mod);//除了这个子
树外的答案重新算
            sum[from][i]=ADD(sum[from][i],delta),ans[u]=ADD(ans[u],delta);//更新答
案
        }nowans=ADD(nowans,
(2ll*val[edge]+addx)*addx%mod*psb[edge]%mod),val[edge]=ADD(val[edge],addx);
        //(w+addx)^2=w^2+addx^2+2*w*addx,w^2已经计算过
    }
    void solve()
    {
        int x,y;
        dfsinit(1,0),msz[root]=nown=n;
        dfsroot(1,0),dfs(root,0);
        write(nowans);
        while(q--)
        {
            read(x),read(y);
            update(x,y),write(nowans);
            if(q%100000==0)flush();
        }flush();
    }
}
int main()
{
    int uzless,x;
    read(uzless);
    read(n),read(q);
    f(i,2,n)read(x),read(val[i]),add(i,x);
    NDC_Tree::solve();
}//代码中有部分其实不会爆ull的未取模(2ll*...)乘法
```

polynomial

```
#define f(i,a,b) for(register int i=(a),i##end=(b);i<=i##end;i=-(~i))
#define fe(i,a,b) for(register int i=(a),i##end=(b);i<i##end;i=-(~i))
#define rf(i,a,b) for(register int i=(a),i##end=(b);i>=i##end;i=~(-i))
const int neko=100010,feko=400010,mod=998244353;
int n,a[neko],fac[neko],ifac[neko];
int grt[60][2];
typedef int arr[feko];
arr A,B,X,F,E,rev,gp;
int ADD(int x,int y){return (x+=y)>=mod?x-mod:x;}
int spow(int m,int n)
{
    int b=1;
    for(;n;n>>=1,m=1ll*m*m%mod)if(n&1)b=1ll*b*m%mod;
    return b;
}
```

```
namespace Conv
{
    using std::swap;
    const int gen=3,igen=spow(3,mod-2);
    void NTT(int *p,int n,int opt)
    {
        fe(i,0,n)if(i<rev[i])swap(p[i],p[rev[i]]);
        int hfi,u,v;
        for(int i=2,cnt=1;i<=n;i<<=1,++cnt)
        {
            hfi=i>>1;
            fe(j,1,hfi)gp[j]=1ll*gp[j-1]*grt[cnt][opt]%mod;
            for(int j=0;j<n;j+=i)
                fe(k,0,hfi)
                {
                    u=p[j+k],v=1ll*p[j+k+hfi]*gp[k]%mod;
                    p[j+k]=ADD(u,v),p[j+k+hfi]=ADD(u,mod-v);
                }
        }
        if(!opt)
        {
            int in=spow(n,mod-2);
            fe(i,0,n)p[i]=1ll*p[i]*in%mod;
        }
    }
    void getrev(int n,int cnt)
    {fe(i,0,n)rev[i]=(rev[i>>1]>>1)|((i&1)<<cnt);}
    void init(int n)
    {gp[0]=1;for(int i=2,cnt=1;i<=n;i<<=1,++cnt)grt[cnt][1]=spow(gen,(mod-
1)/i),grt[cnt][0]=spow(igen,(mod-1)/i);}
}
namespace Poly
{
    using namespace Conv;
    void Inv(int len,int *C,int *X)//len是长度!
    {
        if(len==1){X[0]=spow(C[0],mod-2);return;}
        Inv(len+1>>1,C,X);
        int n,cnt;
        for(n=1,cnt=0;n<(len<<1);n<<=1)++cnt;--cnt;//长度一定不大于len<<1
        fe(i,0,len)A[i]=C[i];fe(i,len,n)A[i]=0;//A在这个模意义下!
        getrev(n,cnt);
        NTT(A,n,1),NTT(X,n,1);
        fe(i,0,n)X[i]=ADD(2ll*X[i]%mod,mod-1ll*A[i]*X[i]%mod*X[i]%mod);
        NTT(X,n,0);
        fe(i,len,n)X[i]=0;
    }
    void Int(int n,int *C)
    {rf(i,n,1)C[i]=1ll*C[i-1]*fac[i-1]%mod*ifac[i]%mod;C[0]=0;}
    void Der(int n,int *C)
    {fe(i,0,n-1)C[i]=1ll*C[i+1]*(i+1)%mod;C[n]=0;}
    void Ln(int len,int *C)
    {
        Inv(len,C,X),Der(len,C);
        int n,cnt;
        for(n=1,cnt=0;n<(len<<1);n<<=1)++cnt;--cnt;
        getrev(n,cnt);
        NTT(C,n,1),NTT(X,n,1);
```

```
        fe(i,0,n)C[i]=1ll*C[i]*X[i]%mod,X[i]=0;
        NTT(C,n,0);
        fe(i,len,n)C[i]=0;
        Int(len,C);
    }
    void Exp(int len,int *C,int *E)
    {
        if(len==1){E[0]=1;return;}
        Exp(len+1>>1,C,E);
        int n,cnt;
        for(n=1,cnt=0;n<(len<<1);n<<=1)++cnt;--cnt;
        getrev(n,cnt);
        fe(i,0,len)F[i]=E[i];Ln(len,F);
        fe(i,0,len)A[i]=ADD(mod-F[i],C[i]);A[0]=ADD(A[0],1);fe(i,len,n)A[i]=0;
        NTT(A,n,1),NTT(E,n,1);
        fe(i,0,n)E[i]=1ll*E[i]*A[i]%mod;
        NTT(E,n,0);
        fe(i,len,n)E[i]=0;
    }
}
int main()
{
    using namespace Poly;
    scanf("%d",&n),--n;
    int x,cnt=0;
    for(x=2;x<(n<<1|1);x<<=1);Conv::init(x);
    f(i,0,n)scanf("%d",&a[i]);
    Inv(n+1,a,X);
    Exp(n+1,a,E);
    f(i,0,n)printf("%d%c",X[i],i^iend?' ':'\n');
    return 0;
}
```

常系数线性递推

```
#include<cstdio>
#include<iostream>
#include<cstring>
#include<algorithm>
#define f(i,a,b) for(register int i=(a);i<=(b);i=-(~i))
#define fe(i,a,b) for(register int i=(a);i<(b);i=-(~i))
const int keko=32010,feko=600010,mod=998244353;
int n,k,a[keko],f[keko],gpow[1010][2],ans;
typedef int arr[feko];
arr C,X,G,F,rev,A,B,Hr,Qr,gi;
//C:c序列 F:特征多项式 G:转移矩阵系数 X:逆元
int ADD(int x,int y)
{return (x+=y)>=mod?x-mod:x;}
int cmin(int x,int y){return x<y?x:y;}
int spow(int m,int n)
{
    int b=1;
    for(;n;n>>=1,m=1ll*m*m%mod)if(n&1)b=1ll*b*m%mod;
    return b;
}
namespace Conv
```

```
{
    using std::swap;
    int m,n;
    const int gen=3,igen=spow(3,mod-2);
    void NTT(int *p,int opt)
    {
        fe(i,0,n)if(i<rev[i])swap(p[i],p[rev[i]]);
        int hfi,u,v;
        for(register int i=2,tp=1;i<=n;i<<=1,++tp)
        {
            hfi=i>>1;
            fe(j,1,hfi)gi[j]=1ll*gi[j-1]*gpow[tp][opt]%mod;
            for(register int j=0;j<n;j+=i)
            {
                fe(k,0,hfi)
                {
                    u=p[j+k],v=1ll*gi[k]*p[j+k+hfi]%mod;
                    p[j+k]=ADD(u,v),p[j+k+hfi]=ADD(u,mod-v);
                }
            }
        }
        if(!opt)
        {
            int in=spow(n,mod-2);
            fe(i,0,n)p[i]=1ll*p[i]*in%mod;
        }
    }
    int solve(int n1,int n2,int opt)
    {
        m=n1+n2;int cnt=0;
        for(n=1;n<=m;n<<=1)++cnt;--cnt;
        if(~cnt)fe(i,0,n)rev[i]=(rev[i>>1]>>1)|((i&1)<<cnt);
        //printf("%d %d\n",opt,n);
        NTT(A,1),NTT(B,1);
        if(opt==1)fe(i,0,n)A[i]=1ll*A[i]*B[i]%mod;
        else fe(i,0,n)A[i]=1ll*A[i]*B[i]%mod*B[i]%mod;
        NTT(A,0);
        return n;
    }
    void init(int n)
    {gi[0]=1;for(register int i=2,tp=1;i<=n;i<<=1,++tp)gpow[tp][1]=spow(gen,(mod-
1)/i),gpow[tp][0]=spow(igen,(mod-1)/i);}
}
namespace Poly
{
    using std::reverse;
    using namespace Conv;
    int flag=0;
    void Mul(int n,int m,int *C,int nx)
    {
        int Cn=solve(n,m,1);
        f(i,0,nx)C[i]=A[i],A[i]=B[i]=0;
        f(i,nx+1,Cn)A[i]=B[i]=0;
    }
    void Inv(int *C,int n)
    {
        int len=2;X[0]=spow(C[0],mod-2);
        while(len<=n)
```

```
        {
            fe(i,0,len>>1)A[i]=C[i],B[i]=X[i];
            solve((len>>1)-1,len-1,2);
            fe(i,0,len)X[i]=ADD(ADD(X[i],X[i]),mod-A[i]);
            fe(i,len,len<<1)A[i]=B[i]=0;
            len<<=1;
        }f(i,0,len)A[i]=B[i]=0;
    }
    void Div(int *H,int *Q,int n,int m)//H=WQ+P
    {
        if(n<m)return;
        memcpy(Hr,H,sizeof Hr);
        reverse(Hr,Hr+n+1);
        f(i,n-m+1,n)Hr[i]=0;
        //printf("%d\n",n);
        /*puts("orz?");
        f(i,0,x)printf("%d ",X[i]);
        putchar('\n');*/
        f(i,0,n-m)A[i]=Hr[i],B[i]=X[i];
        Mul(n-m,n-m,Hr,n-m);
        reverse(Hr,Hr+n-m+1);
        f(i,0,n-m)A[i]=Hr[i],B[i]=Q[i];
        f(i,n-m+1,m)B[i]=Q[i];
        solve(n-m,m,1);
        fe(i,0,m)H[i]=ADD(H[i],mod-A[i]);
        f(i,m,n)H[i]=0;
        memset(A,0,sizeof A),memset(B,0,sizeof B);
    }
    void Mod(int *C,int *D,int *X,int n)
    {
        fe(i,0,n)A[i]=C[i],B[i]=D[i];
        Mul(n-1,n-1,C,n-1);
        Div(C,X,n,k);
    /*  fe(i,0,k)printf("%d ",C[i]);
        putchar('\n');*/
    }
    void Spow(int *M,int n,int *B,int *X)
    {
        int y;
        y=k<<1;
        for(register int x=1;n;n>>=1,Mod(M,M,X,2<<x),++x)
        {
            if(n&1)Mod(B,M,X,y);
            if((1<<(x-1))>=y)--x;
        }
    }
    void Init(int n,int m)
    {
        memcpy(Qr,F,sizeof Qr);
        reverse(Qr,Qr+m+1);
        int x;
        for(x=2;x<(n-m+1);x<<=1);
        Inv(Qr,x);
    }
}
int main()
{
    scanf("%d%d",&n,&k);
```

```
    int x;
    for(x=2;x<=k;x<<=1);
    Conv::init(x<<3);
    //让我们算一笔账：求逆<<2，快速幂的2<<x最大可以到y(k<<1)的两倍，两者都需要再<<1（因为是乘法），于是就<<3(*8)
    f(i,1,k)scanf("%d",&f[i]),F[k-i]=ADD(0,mod-f[i]);
    F[k]=1;
    fe(i,0,k)scanf("%d",&a[i]),a[i]=ADD(a[i],mod);
    G[1]=1,C[0]=1;//实际上是求转移矩阵的系数，也就是我们把转移矩阵设成一个未知数x，它是一次的，系数是1
    Poly::Init(k<<2,k);//只用求一次逆元
    Poly::Spow(G,n,C,F);
    fe(i,0,k)ans=ADD(1ll*C[i]*a[i]%mod,ans);
    return printf("%d\n",ADD(ans,mod)),std::cerr<<clock()*1.0/CLOCKS_PER_SEC<<std::endl,0;
}
```

SAM

```
namespace SAM {
    int cnt=0,las=0,cur;
    void extend(char *s,int n)
    {
        int p,q,clone,x;
        link[0]=-1;
        f(i,1,n)
        {
            x=s[i]-'a';
            cur=++cnt,len[cur]=len[las]+1,res[cur]=num[n-i+2];
            for(p=las;p!=-1&&(!nex[p][x]);p=link[p])nex[p][x]=cur;
            if(p==-1)link[cur]=0;
            else
            {
                q=nex[p][x];
                if(len[q]==len[p]+1)link[cur]=q;
                else
                {
                    clone=++cnt,len[clone]=len[p]+1;
                    link[clone]=link[q];
                    memcpy(nex[clone],nex[q],sizeof nex[q]);
                    for(;p!=-1&&(nex[p][x]==q);p=link[p])nex[p][x]=clone;
                    link[cur]=link[q]=clone;
                }
            }las=cur;
        }
    }
    void linktree()
    {
        f(i,0,cnt)vec[len[i]].pb(i);
        rf(i,neko-5,1)for(int x:vec[i])res[link[x]]+=res[x];
    }
}
```

莫队

```
//luogu P1494 小Z的袜子.具体内容详见博客.
#define f(i,a,b) for(register int i=(a),i##end=(b);i<=i##end;i=-(~i))
const int neko=50010;
int n,m,col[neko],cnt[neko],bl[neko],blk;
typedef std::pair<long long,long long> pi;
pi ans[neko];
int gcd(int a,int b){return b?gcd(b,a%b):a;}
struct node
{
    int l,r,id;
    bool operator <(const node &x)const
    {
        if(bl[l]==bl[x.l])
        {
            if(bl[l]&1)return r<x.r;
            return r>x.r;
        }return bl[l]<bl[x.l];
    }
}q[neko];
namespace Mo
{
    pi now=pi(0,0);int l,r;
    void add(int x,int opt)
    {
        now.se+=opt*(r-l);
        if(opt==1)now.fi+=cnt[col[x]],++cnt[col[x]];
        else --cnt[col[x]],now.fi-=cnt[col[x]];
    }
}
int main()
{
    using namespace Mo;
    int x;
    scanf("%d%d",&n,&m),blk=n/sqrt(m);
    f(i,1,n)scanf("%d",&col[i]);
    f(i,1,m)scanf("%d%d",&q[i].l,&q[i].r),q[i].id=i;
    f(i,1,n)bl[i]=(i-1)/blk+1;
    std::sort(q+1,q+m+1);
    l=1,r=0;
    f(i,1,m)
    {
        while(r<q[i].r)add(++r,1);
        while(r>q[i].r)add(r,-1),--r;
        while(l>q[i].l)add(--l,1);
        while(l<q[i].l)add(l,-1),++l;
        //printf("%d %d %lld\n",l,r,now.se);
        if(q[i].l==q[i].r)ans[q[i].id].fi=0,ans[q[i].id].se=1;
        else
x=gcd(now.fi,now.se),ans[q[i].id].fi=now.fi/x,ans[q[i].id].se=now.se/x;
    }f(i,1,m)printf("%lld/%lld\n",ans[i].fi,ans[i].se);
}
```

回滚莫队

```
//JOISC2014 Day1 历史研究.具体内容详见博客
#include<cstdio>
#include<iostream>
#include<cmath>
#include<algorithm>
#define f(i,a,b) for(register int i=(a),i##end=(b);i<=i##end;i=-(~i))
using namespace std;
const int neko=100010;
int n,m,tp;
typedef int arr[neko];
arr a,b,bl,L,R,cnt,val,s;
typedef long long ll;
ll ans[neko],now,las;
struct qwq
{
    int l,r,id;
    bool operator <(const qwq &x)const
    {return bl[l]==bl[x.l]?r<x.r:l<x.l;}
}q[neko];
int blk;
ll cmax(ll x,ll y){return x>y?x:y;}
namespace Mo
{
    void build()
    {
        f(i,1,n)
        {
            bl[i]=(i-1)/blk+1;
            if(bl[i]^bl[i-1])L[bl[i]]=i,R[bl[i-1]]=i-1;
        }R[bl[n]]=n;
    }
    void add(int x)
    {++cnt[a[x]],now=cmax(now,1ll*cnt[a[x]]*val[a[x]]);}
}
int main()
{
    using namespace Mo;
    int x,l,r;
    scanf("%d%d",&n,&m),blk=n/sqrt(m);
    f(i,1,n)scanf("%d",&a[i]),b[i]=a[i];
    sort(b+1,b+n+1),x=unique(b+1,b+n+1)-(b+1);
    f(i,1,x)val[i]=b[i];f(i,1,n)a[i]=lower_bound(b+1,b+x+1,a[i])-b;
    f(i,1,m)scanf("%d%d",&q[i].l,&q[i].r),q[i].id=i;
    build();
    sort(q+1,q+m+1);
    f(i,1,m)
    {
        if(bl[q[i].l]^bl[q[i-1].l])
        {f(j,1,x)cnt[j]=0;now=las=0,r=R[bl[q[i].l]];}
        if(bl[q[i].l]==bl[q[i].r])
        {
            f(j,q[i].l,q[i].r)add(j),s[++tp]=a[j];
            ans[q[i].id]=now;
        }
        else
```

```
        {
            l=R[bl[q[i].l]]+1;
            while(q[i].r>r)add(++r),las=now;
            while(q[i].l<l)add(--l),s[++tp]=a[l];
            ans[q[i].id]=now;
        }
        while(tp)--cnt[s[tp]],--tp;now=las;
    }f(i,1,m)printf("%lld\n",ans[i]);
    return 0;
}
```

无源汇上下界可行流

```
#define f(i,a,b) for(register int i=(a),i##end=(b);i<=i##end;i=-(~i))
using namespace std;
const int neko=210,ceko=30600*2+10,inf=2e9;
int n,m,t=1;
typedef int arr[neko];
arr head,cur,dis;
struct node
{int v,nex,cap;}e[ceko];
int lowf[ceko];
int cmin(int x,int y){return x<y?x:y;}
void add(int x,int y,int z,int o)
{
    e[++t].v=y,e[t].cap=z,lowf[t]=o,e[t].nex=head[x],head[x]=t;
    e[++t].v=x,e[t].cap=0,lowf[t]=o,e[t].nex=head[y],head[y]=t;
}
namespace Flow
{
    int SS,TT;
    bool bfs(int S,int T)
    {
        int u;
        queue<int>q;
        memset(dis,0,sizeof dis);
        q.push(S),dis[S]=1;
        while(!q.empty())
        {
            u=q.front(),q.pop();
            for(register int
i=head[u],v=e[i].v;i;i=e[i].nex,v=e[i].v)if(!dis[v]&&e[i].cap)
            {
                dis[v]=dis[u]+1;
                q.push(v);
                if(v==T)return 1;
            }
        }return 0;
    }
    int dfs(int u,int flow,int T)
    {
        int up,rescap=0;
        if(!flow||u==T)return flow;
        for(register int
&i=cur[u],v=e[i].v;i;i=e[i].nex,v=e[i].v)if(dis[v]==dis[u]+1&&
(up=dfs(v,cmin(e[i].cap,flow),T)))
        {
```

```
            e[i].cap-=up,e[i^1].cap+=up;
            rescap+=up;
            if(!(flow-=up))break;
        }return rescap;
    }
    int dinic(int S,int T)
    {
        int ans=0;
        while(bfs(S,T))memcpy(cur,head,sizeof cur),ans+=dfs(S,inf,T);
        return ans;
    }
}
int main()
{
    using namespace Flow;
    int res=0,x,y,z,o;
    scanf("%d%d",&n,&m),SS=n+1,TT=SS+1;
    f(i,1,m)scanf("%d%d%d%d",&x,&y,&z,&o),add(x,y,o-
z,z),add(SS,y,z,0),add(x,TT,z,0),res+=z;
    if(res^dinic(SS,TT))printf("NO\n");
    else
    {
        printf("YES\n");

f(i,2,t)if((i&1)&&e[i].v<=n&&e[i].v>=1&&e[i^1].v<=n&&e[i^1].v>=1)printf("%d\n",e[
i].cap+lowf[i]);//反向边残量=流量
    }
    return 0;
}
```

有源汇上下界最大流

```
const int neko=210,meko=30010,inf=2147483647;
int n,m,t=1;
struct node
{int v,cap,nex;}e[meko<<1];
int dis[neko],head[neko],cur[neko];
int cmin(int x,int y){return x<y?x:y;}
void add(int x,int y,int z)
{
    e[++t].v=y,e[t].cap=z,e[t].nex=head[x],head[x]=t;
    e[++t].v=x,e[t].cap=0,e[t].nex=head[y],head[y]=t;
}
namespace Dinic
{
    int S,T,SS,TT;
    bool bfs(int S,int T)
    {
        std::queue<int>q;
        memset(dis,0,sizeof dis);
        q.push(S),dis[S]=1;
        int u;
        while(!q.empty())
        {
            u=q.front(),q.pop();
            for(int
i=head[u],v=e[i].v;i;i=e[i].nex,v=e[i].v)if(e[i].cap&&!dis[v])
```

```
            {
                dis[v]=dis[u]+1;
                if(v==T)return 1;
                q.push(v);
            }
        }return 0;
    }
    int dfs(int u,int T,int flow)
    {
        if(u==T||!flow)return flow;
        int up,rescap=0;
        for(int &i=cur[u],v=e[i].v;i;i=e[i].nex,v=e[i].v)if(dis[v]==dis[u]+1&&
(up=dfs(v,T,cmin(flow,e[i].cap))))
        {
            e[i].cap-=up,e[i^1].cap+=up;
            rescap+=up;
            if(!(flow-=up))break;
        }return rescap;
    }
    int dinic(int S,int T)
    {
        int ans=0;
        while(bfs(S,T))memcpy(cur,head,sizeof cur),ans+=dfs(S,T,inf);
        return ans;
    }
}
int main()
{
    int x,y,z,o,res=0;
    using namespace Dinic;
    scanf("%d%d%d%d",&n,&m,&S,&T),SS=n+1,TT=SS+1;
    f(i,1,m)scanf("%d%d%d%d",&x,&y,&z,&o),add(x,y,o-
z),add(SS,y,z),add(x,TT,z),res+=z;
    add(T,S,inf);
    if(dinic(SS,TT)^res)return printf("please go home to sleep\n"),0;
    res=e[t].cap,e[t].cap=e[t^1].cap=0,printf("%d\n",res+dinic(S,T));
    return 0;
}
```

有源汇上下界最小流

```
const int neko=50010,meko=375010,inf=2147483647;
int n,m,t=1;
struct node
{int v,cap,nex;}e[meko<<1];
int dis[neko],head[neko],cur[neko];
int cmin(int x,int y){return x<y?x:y;}
void add(int x,int y,int z)
{
    e[++t].v=y,e[t].cap=z,e[t].nex=head[x],head[x]=t;
    e[++t].v=x,e[t].cap=0,e[t].nex=head[y],head[y]=t;
}
namespace Dinic
{
    int S,T,SS,TT;
    bool bfs(int S,int T)
    {
```

```
        std::queue<int>q;
        memset(dis,0,sizeof dis);
        q.push(S),dis[S]=1;
        int u;
        while(!q.empty())
        {
            u=q.front(),q.pop();
            for(int
i=head[u],v=e[i].v;i;i=e[i].nex,v=e[i].v)if(e[i].cap&&!dis[v])
            {
                dis[v]=dis[u]+1;
                if(v==T)return 1;
                q.push(v);
            }
        }return 0;
    }
    int dfs(int u,int T,int flow)
    {
        if(u==T||!flow)return flow;
        int up,rescap=0;
        for(int &i=cur[u],v=e[i].v;i;i=e[i].nex,v=e[i].v)if(dis[v]==dis[u]+1&&
(up=dfs(v,T,cmin(flow,e[i].cap))))
        {
            e[i].cap-=up,e[i^1].cap+=up;
            rescap+=up;
            if(!(flow-=up))break;
        }return rescap;
    }
    int dinic(int S,int T)
    {
        int ans=0;
        while(bfs(S,T))memcpy(cur,head,sizeof cur),ans+=dfs(S,T,inf);
        return ans;
    }
}
int main()
{
    int x,y,z,o,res=0;
    using namespace Dinic;
    scanf("%d%d%d%d",&n,&m,&S,&T),SS=n+1,TT=SS+1;
    f(i,1,m)scanf("%d%d%d%d",&x,&y,&z,&o),add(x,y,o-
z),add(SS,y,z),add(x,TT,z),res+=z;
    add(T,S,inf);
    if(dinic(SS,TT)^res)return printf("please go home to sleep\n"),0;//无解
    res=e[t].cap,e[t].cap=e[t^1].cap=0,printf("%d\n",res-dinic(T,S));
    return 0;
}
```