ACM 板子整理

1.数据结构

1.1平衡树

fhq -treap (例子为处理区间翻转)

```
mt19937 rnd(time(0));
int n,m,Rt,tot=0,s[maxn],tag[maxn],x,y,ls[maxn],rs[maxn],w[maxn],v[maxn];
int ins(int x){s[++tot]=1,v[tot]=x,w[tot]=rnd();return tot;}
void push(int rt)
{if(tag[rt])swap(ls[rt],rs[rt]),tag[ls[rt]]^=1,tag[rs[rt]]^=1;tag[rt]=0;}
void pushup(int rt){s[rt]=s[ls[rt]]+s[rs[rt]]+1;}
int merge(int a,int b)
{
    if(!a||!b)return a+b;
    if(w[a]>w[b]){push(b),ls[b]=merge(a,ls[b]);pushup(b);return b;}
    push(a),rs[a]=merge(rs[a],b);pushup(a);return a;
}
void split(int rt,int &a,int &b,int k)
{
    if(!rt){a=0,b=0;return;}push(rt);
    if(s[ls[rt]]>=k)b=rt,split(ls[rt],a,ls[rt],k);
    else a=rt,split(rs[rt],rs[rt],b,k-s[ls[rt]]-1);
    pushup(rt);
}
void change(int l,int r)
{
    int a,b,c;split(Rt,a,b,l-1);
    split(b,b,c,r-l+1);tag[b]^=1;
    Rt=merge(a,merge(b,c));
}
void dfs(int rt)
{
    if(!rt)return;push(rt);
    dfs(ls[rt]);printf("%d ",v[rt]);
    dfs(rs[rt]);
}
signed main()
{
    //freopen(".in","r",stdin);
```

```
    //freopen(".out","w",stdout);
    n=read();m=read();
    for(int i=1;i<=n;i++)Rt=merge(Rt,ins(i));
    for(int i=1;i<=m;i++)
    {
        x=read(),y=read();
        change(x,y);
    }
    dfs(Rt);
    return 0;
}
```

1.2线段树分裂

给出一个可重集 a（编号为 1），它支持以下操作：

`0 p x y`：将可重集 p 中大于等于 x 且小于等于 y 的值移动到一个新的可重集中（新可重集编号为从 2 开始的正整数，是上一次产生的新可重集的编号+1）。

`1 p t`：将可重集 t 中的数放入可重集 p，且清空可重集 t（数据保证在此后的操作中不会出现可重集 t）。

`2 p x q`：在 p 这个可重集中加入 x 个数字 q。

`3 p x y`：查询可重集 p 中大于等于 x 且小于等于 y 的值的个数。

`4 p k`：查询在 p 这个可重集中第 k 小的数，不存在时输出 `-1`。

```
#include<bits/stdc++.h>
using namespace std;
#define int long long
const int maxn=2e5+5;
inline int read()
{
    char ch=getchar();bool f=0;int x=0;
    for(;!isdigit(ch);ch=getchar())if(ch=='-')f=1;
    for(;isdigit(ch);ch=getchar())x=(x<<1)+(x<<3)+(ch^48);
    if(f==1)x=-x;return x;
}
int v[maxn],son[maxn<<5][2],s[maxn<<5],opt,x,y,rt[maxn],tot=0,n,m,sum=1,z;
void modify(int &Rt,int l,int r,int x,int num)
{
    if(l>x||r<x)return;
    if(!Rt)Rt=++tot;s[Rt]+=num;
    if(l==r)return;int mid=l+r>>1;
    modify(son[Rt][0],l,mid,x,num);modify(son[Rt][1],mid+1,r,x,num);
}
int query(int Rt,int l,int r,int L,int R)
{
    if(R<l||L>r)return 0;
    if(l>=L&&r<=R)return s[Rt];
    int mid=l+r>>1;
    return query(son[Rt][0],l,mid,L,R)+query(son[Rt][1],mid+1,r,L,R);
}
int getkth(int Rt,int l,int r,int k)
```

```
{
    if(l==r)return l;int mid=l+r>>1;
    if(s[son[Rt][0]]>=k)return getkth(son[Rt][0],l,mid,k);
    return getkth(son[Rt][1],mid+1,r,k-s[son[Rt][0]]);
}
void split(int &x,int &y,int k)
{
    if(!x)return;y=++tot;
    if(s[son[x][0]]<k)split(son[x][1],son[y][1],k-s[son[x][0]]);
    else son[y][1]=son[x][1],son[x][1]=0;
    if(s[son[x][0]]>k)split(son[x][0],son[y][0],k);s[y]=s[x]-k;s[x]=k;
}
int merge(int x,int y){
    if(!x||!y) return x+y;s[x]+=s[y];
    son[x][0]=merge(son[x][0],son[y][0]),son[x][1]=merge(son[x][1],son[y][1]);
    son[y][0]=son[y][1]=s[y]=0;return x;
}
signed main()
{
    n=read(),m=read();
    for(int i=1;i<=n;i++)x=read(),modify(rt[1],1,n,i,x);
    for(int i=1;i<=m;i++){
        opt=read();x=read(),y=read();
        if(opt==0){
            z=read();int k1=query(rt[x],1,n,1,z),k2=query(rt[x],1,n,y,z);
            int rt1=0;split(rt[x],rt[++sum],k1-k2);split(rt[sum],rt1,k2);
            rt[x]=merge(rt[x],rt1);
        }
        if(opt==1)rt[x]=merge(rt[x],rt[y]);
        if(opt==2)z=read(),modify(rt[x],1,n,z,y);
        if(opt==3)z=read(),printf("%lld\n",query(rt[x],1,n,y,z));
        if(opt==4){
            if(y>s[rt[x]])puts("-1");
            else printf("%lld\n",getkth(rt[x],1,n,y));
        }
    }
    return 0;
}
```

1.3可并堆（左偏树）

一开始有 n$个小根堆，每个堆包含且仅包含一个数。接下来需要支持两种操作：

1. `1 x y`：将第 x 个数和第 y 个数所在的小根堆合并（若第 x 或第 y 个数已经被删除或第 x 和第 y 个数在同一个堆内，则无视此操作）。
2. `2 x`：输出第 x 个数所在的堆最小数，并将这个最小数删除（若有多个最小数，优先删除先输入的；若第 x 个数已经被删除，则输出 -1 并无视删除操作）。

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=1e5+5;
#define ls c[x][0]
#define rs c[x][1]
inline int read()
```

```
{
    char ch=getchar();bool f=0;int x=0;
    for(;!isdigit(ch);ch=getchar())if(ch=='-')f=1;
    for(;isdigit(ch);ch=getchar())x=(x<<1)+(x<<3)+(ch^48);
    if(f==1)x=-x;return x;
}
int fa[maxn],v[maxn],dis[maxn],c[maxn][2],n,m,op,x,y;
int find(int x){if(fa[x]==x)return x;return fa[x]=find(fa[x]);}
int merge(int x,int y){
    if(!x||!y)return x+y;
    if(v[x]>v[y]||(v[x]==v[y]&&x>y))swap(x,y);
    rs=merge(rs,y);
    if(dis[rs]>dis[ls])swap(ls,rs);
    dis[x]=dis[rs]+1;fa[x]=fa[ls]=fa[rs]=x;return x;
}
void del(int x){v[x]=-1;fa[x]=fa[ls]=fa[rs]=merge(ls,rs);}
signed main(){
    n=read(),m=read();dis[0]=-1;
    for(int i=1;i<=n;i++)v[i]=read(),fa[i]=i;
    for(int i=1;i<=m;i++){
        op=read();
        if(op==1){
            x=read(),y=read();if(v[x]==-1||v[y]==-1)continue;
            x=find(x),y=find(y);if(x!=y)fa[x]=fa[y]=merge(x,y);
        }
        else {
            x=read();if(v[x]==-1){puts("-1");continue;}
            x=find(x);printf("%d\n",v[x]);del(x);
        }
    }
    return 0;
}
```

1.4动态树

给定 n 个点以及每个点的权值，要你处理接下来的 m 个操作。
操作有四种，操作从 0 到 3 编号。点从 1 到 n 编号。

- `0 x y` 代表询问从 x 到 y 的路径上的点的权值的 xor 和。保证 x 到 y 是联通的。
- `1 x y` 代表连接 x 到 y，若 x 到 y 已经联通则无需连接。
- `2 x y` 代表删除边 (x,y)，不保证边 (x,y) 存在。
- `3 x y` 代表将点 x 上的权值变成 y。

```
#include<bits/stdc++.h>
using namespace std;
#define ls c[x][0]
#define rs c[x][1]
#define isrt(x) (c[fa[x]][0]!=x&&c[fa[x]][1]!=x)
#define get(x) (c[fa[x]][1]==x)
const int maxn=2e5+5;
inline int read(){
    char ch=getchar();bool f=0;int x=0;
    for(;!isdigit(ch);ch=getchar())if(ch=='-')f=1;
```

```
    for(;isdigit(ch);ch=getchar())x=(x<<1)+(x<<3)+(ch^48);
    if(f==1)x=-x;return x;
}
int a[maxn],n,m,x,y,s[maxn],st[maxn],op,fa[maxn],c[maxn][2],r[maxn];
map<int,int>p[maxn];
void pushup(int x){s[x]=s[ls]^s[rs]^a[x];}
void pushx(int x){if(!x)return ;r[x]^=1;swap(ls,rs);}
void push(int x){if(r[x])pushx(ls),pushx(rs);r[x]=0;}
void rot(int x){
    int y=fa[x],z=fa[fa[x]];int s1=get(x),s2=get(y),w=c[x][!s1];
    if(!isrt(y))c[z][s2]=x;fa[x]=z;
    c[x][!s1]=y;fa[y]=x;c[y][s1]=w;if(w)fa[w]=y;pushup(y);pushup(x);
}
void splay(int x){
    int top=0,y=x;st[++top]=x;
    while(!isrt(y))y=fa[y],st[++top]=y;
    while(top)push(st[top]),top--;
    while(!isrt(x)){
        if(!isrt(fa[x]))rot((get(x)==get(fa[x]))?fa[x]:x);
        rot(x);
    }
}
void access(int x){for(int y=0;x;y=x,x=fa[x])splay(x),rs=y,pushup(x);}
void makert(int x){access(x),splay(x),pushx(x);}
void split(int x,int y){makert(x),access(y);splay(y);}
int findrt(int x){access(x);splay(x);while(ls){push(x),x=ls;}splay(x);return x;}
void link(int x,int y){makert(x);if(findrt(y)!=x)fa[x]=y,p[x][y]=p[y][x]=1;}
void cut(int x,int y){if(!p[x][y])return;p[x][y]=p[y][x]=0;makert(x);int
z=findrt(y);fa[y]=c[x][1]=0,pushup(x);}
signed main(){
    n=read(),m=read();
    for(int i=1;i<=n;i++)a[i]=read();
    for(int i=1;i<=m;i++){
        op=read();x=read(),y=read();
        if(op==0)split(x,y),printf("%d\n",s[y]);
        if(op==1)link(x,y);
        if(op==2)cut(x,y);
        if(op==3)splay(x),a[x]=y;

    }
    return 0;
}
```

1.5树链剖分

如题，已知一棵包含 N 个结点的树（连通且无环），每个节点上包含一个数值，需要支持以下操作：

- `1 x y z`，表示将树从 x 到 y 结点最短路径上所有节点的值都加上 z。
- `2 x y`，表示求树从 x 到 y 结点最短路径上所有节点的值之和。
- `3 x z`，表示将以 x 为根节点的子树内所有节点值都加上 z。
- `4 x` 表示求以 x 为根节点的子树内所有节点值之和

```
#include<bits/stdc++.h>
using namespace std;
```

```
#define pi pair<int,int>
#define mp make_pair
#define fi first
#define se second
#define pb push_back
#define ls (rt<<1)
#define rs (rt<<1|1)
#define mid (l+r>>1)
const int maxn=1e5+5;
inline int read(){
    char ch=getchar();bool f=0;int x=0;
    for(;!isdigit(ch);ch=getchar())if(ch=='-')f=1;
    for(;isdigit(ch);ch=getchar())x=(x<<1)+(x<<3)+(ch^48);
    if(f==1)x=-x;return x;
}
inline void print(int x){
    static int a[55];int top=0;
    if(x<0) putchar('-'),x=-x;
    do{a[top++]=x%10,x/=10;}while(x);
    while(top) putchar(a[--top]+48);
}
vector<int>e[maxn];
int a[maxn],n,m,mod,g,op,x,y,cnt=0,z;
int sz[maxn],son[maxn],fa[maxn],d[maxn],top[maxn],dfn[maxn],idx=0,b[maxn];
int sum[maxn*4],tag[maxn*4];
void dfs1(int x){
    d[x]=d[fa[x]]+1;sz[x]=1;
    for(auto i:e[x])
        if(i!=fa[x]){
            fa[i]=x;dfs1(i);sz[x]+=sz[i];
            if(sz[son[x]]<sz[i])son[x]=i;
        }
}
void dfs2(int x,int tf){
    dfn[x]=++idx;b[idx]=a[x];top[x]=tf;
    if(son[x])dfs2(son[x],tf);
    for(auto i:e[x])if(i!=fa[x]&&i!=son[x])dfs2(i,i);
}
void pushx(int rt,int x,int y){tag[rt]+=x;sum[rt]=(sum[rt]+x*y)%mod;}
void push(int rt,int l,int r){pushx(ls,tag[rt],mid-l+1);pushx(rs,tag[rt],r-
mid);tag[rt]=0;}
void build(int rt,int l,int r){
    if(l==r){sum[rt]=b[l];return;}
    build(ls,l,mid),build(rs,mid+1,r);
    sum[rt]=(sum[ls]+sum[rs])%mod;
}
void add(int rt,int l,int r,int L,int R,int num){
    if(l>R||r<L)return;
    if(l>=L&&r<=R){pushx(rt,num,r-l+1);return;}
    push(rt,l,r);add(ls,l,mid,L,R,num),add(rs,mid+1,r,L,R,num);
    sum[rt]=(sum[ls]+sum[rs])%mod;
}
int query(int rt,int l,int r,int L,int R){
    if(l>R||r<L)return 0;
    if(l>=L&&r<=R)return sum[rt];push(rt,l,r);
    return (query(ls,l,mid,L,R)+query(rs,mid+1,r,L,R))%mod;
```

```
}
void treeadd(int x,int y,int num){
    while(top[x]!=top[y]){
        if(d[top[x]]<d[top[y]])swap(x,y);
        add(1,1,n,dfn[top[x]],dfn[x],num);
        x=fa[top[x]];
    }if(d[x]>d[y])swap(x,y);
    add(1,1,n,dfn[x],dfn[y],z);
}
int treequery(int x,int y){
    int res=0;
    while(top[x]!=top[y]){
        if(d[top[x]]<d[top[y]])swap(x,y);
        res=(res+query(1,1,n,dfn[top[x]],dfn[x]))%mod;
        x=fa[top[x]];
    }if(d[x]>d[y])swap(x,y);
    res=(res+query(1,1,n,dfn[x],dfn[y]))%mod;
    return res;
}
int main()
{
    n=read(),m=read(),g=read(),mod=read();
    for(int i=1;i<=n;i++)a[i]=read();
    for(int i=1;i<n;i++)x=read(),y=read(),e[x].pb(y),e[y].pb(x);
    dfs1(g),dfs2(g,g);build(1,1,n);
    for(int i=1;i<=m;i++){
        op=read();
        if(op==1)x=read(),y=read(),z=read(),treeadd(x,y,z);
        if(op==2)x=read(),y=read(),printf("%d\n",treequery(x,y));
        if(op==3)x=read(),z=read(),add(1,1,n,dfn[x],dfn[x]+sz[x]-1,z);
        if(op==4)x=read(),printf("%d\n",query(1,1,n,dfn[x],dfn[x]+sz[x]-1));
    }
    return 0;
}
```

1.6动态dp 动态树分治

给定一棵 n 个点的树，点带点权。

有 m 次操作，每次操作给定 x, y，表示修改点 x 的权值为 y。

你需要在每次操作之后求出这棵树的最大权独立集的权值大小。

```
#include<bits/stdc++.h>
using namespace std;
#define pi pair<int,int>
#define mp make_pair
#define fi first
#define se second
#define pb push_back
#define mid (l+r>>1)
#define lowbit(x) (x&-x)
const int maxn=1e5+5,M=34005,inf=0x3f3f3f3f;
inline int read()
{
    char ch=getchar();bool f=0;int x=0;
```

```
    for(;!isdigit(ch);ch=getchar())if(ch=='-')f=1;
    for(;isdigit(ch);ch=getchar())x=(x<<1)+(x<<3)+(ch^48);
    if(f==1)x=-x;return x;
}
inline void print(int x)
{
    static int a[55];int top=0;
    if(x<0) putchar('-'),x=-x;
    do{a[top++]=x%10,x/=10;}while(x);
    while(top) putchar(a[--top]+48);
}
int n,m,x,y,v[maxn],a[maxn],sz[maxn],son[maxn];
vector<int>e[maxn];
struct Mat
{
    int a[2][2];Mat(){a[0][0]=a[0][1]=a[1][0]=a[1][1]=-inf;}
    Mat(int x){a[0][0]=a[1][1]=0;a[1][0]=a[0][1]=-inf;}
    friend Mat operator *(Mat a,Mat b)
    {
        Mat c;
        for(int i=0;i<2;i++)
            for(int j=0;j<2;j++)
                for(int k=0;k<2;k++)
                    c.a[i][k]=max(c.a[i][k],a.a[i][j]+b.a[j][k]);
        return c;
    }
    int getMax(){return max(max(a[0][0],a[0][1]),max(a[1][0],a[1][1]));}
    int* operator [](const int &x){return a[x];}
};
struct BST
{
    int ls[maxn],rs[maxn],fa[maxn],st[maxn],top,lsz[maxn],Rt;
    bool vis[maxn];Mat mul[maxn],v[maxn];
    void pushup(int rt){mul[rt]=mul[ls[rt]]*v[rt]*mul[rs[rt]];}
    void getv(int x,int y)
    {v[x][1][0]+=mul[y].getMax();v[x][0][0]=v[x][1][0];
    v[x][0][1]+=max(mul[y][0][0],mul[y][1][0]);fa[y]=x;}
    void init(){v[0]=mul[0]=Mat(1);for(int i=1;i<=n;i++)v[i][0][1]=a[i],v[i][0]
[0]=v[i][1][0]=0;}
    int sbuild(int l,int r)//对序列建bst,找重心递归建树
    {
        if(l>r)return 0;int tot=0;
        for(int i=l;i<=r;i++)tot+=lsz[st[i]];
        for(int i=l,tmp=lsz[st[i]];i<=r;i++,tmp+=lsz[st[i]])
            if(tmp*2>=tot)
            {
                rs[st[i]]=sbuild(l,i-1),ls[st[i]]=sbuild(i+1,r);
                fa[ls[st[i]]]=st[i],fa[rs[st[i]]]=st[i];pushup(st[i]);
                return st[i];
            }
    }
    int build(int x)//链分治，每次处理一条链
    {
        for(int i=x;i;i=son[i])vis[i]=1;
        for(int i=x;i;i=son[i])
            for(auto j:e[i])if(!vis[j])getv(i,build(j));top=0;
```

```
        for(int i=x;i;i=son[i])st[++top]=i,lsz[i]=sz[i]-sz[son[i]];return
sbuild(1,top);
    }
    bool check(int x){return (ls[fa[x]]!=x)&&(rs[fa[x]]!=x)&&fa[x];}//判断是不是轻
边
    void change(int rt,int x)
    {
        v[rt][0][1]+=x-a[rt];a[rt]=x;
        for(int i=rt;i;i=fa[i])
            if(check(i))
            {
                //cout<<i<<endl;
                v[fa[i]][0][0]-=mul[i].getMax();v[fa[i]][1][0]=v[fa[i]][0][0];
                v[fa[i]][0][1]-=max(mul[i][0][0],mul[i][1][0]);pushup(i);
                v[fa[i]][0][0]+=mul[i].getMax();v[fa[i]][1][0]=v[fa[i]][0][0];
                v[fa[i]][0][1]+=max(mul[i][0][0],mul[i][1][0]);
            }else pushup(i);
    }
}T;
void dfs(int x,int fa)
{
    sz[x]=1;
    for(auto i:e[x])
        if(i^fa){dfs(i,x);sz[x]+=sz[i];if(sz[son[x]]<sz[i])son[x]=i;}
}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    n=read(),m=read();
    for(int i=1;i<=n;i++)a[i]=read();
    for(int i=1;i<n;i++)x=read(),y=read(),e[x].pb(y),e[y].pb(x);
    dfs(1,0);T.init();T.Rt=T.build(1);
    for(int i=1;i<=m;i++)
    {
        x=read(),y=read();T.change(x,y);
        printf("%d\n",T.mul[T.Rt].getMax());
    }
    return 0;
}
```

1.7树套树

您需要写一种数据结构（可参考题目标题），来维护一个有序数列，其中需要提供以下操作：

1. 查询 k 在区间内的排名
2. 查询区间内排名为 k 的值
3. 修改某一位置上的数值
4. 查询 k 在区间内的前驱（前驱定义为严格小于 x，且最大的数，**若不存在输出** `-2147483647`）
5. 查询 k 在区间内的后继（后继定义为严格大于 x，且最小的数，**若不存在输出** `2147483647`）

```
#include<bits/stdc++.h>
```

```
using namespace std;
#define pi pair<int,int>
#define mp make_pair
#define fi first
#define se second
#define pb push_back
#define mid (l+r>>1)
#define lowbit(x) (x&-x)
const int maxn=1e5+5,M=34005,inf=1e8+5;
inline int read()
{
    char ch=getchar();bool f=0;int x=0;
    for(;!isdigit(ch);ch=getchar())if(ch=='-')f=1;
    for(;isdigit(ch);ch=getchar())x=(x<<1)+(x<<3)+(ch^48);
    if(f==1)x=-x;return x;
}
inline void print(int x)
{
    static int a[55];int top=0;
    if(x<0) putchar('-'),x=-x;
    do{a[top++]=x%10,x/=10;}while(x);
    while(top) putchar(a[--top]+48);
}
int
n,m,op,l,r,x,ls[maxn*202],rs[maxn*202],sum[maxn*202],a[maxn],rt[maxn],tot=0;vecto
r<int>tmp,tmp2;
void change(int &rt,int l,int r,int pos,int num){
    if(l>pos||r<pos)return;
    if(!rt)rt=++tot;sum[rt]+=num;if(l==r)return;
    change(ls[rt],l,mid,pos,num),change(rs[rt],mid+1,r,pos,num);
}
int query(int rt,int l,int r,int L,int R){
    if(l>R||r<L)return 0;if(!rt)return 0;
    if(l>=L&&r<=R)return sum[rt];
    return query(ls[rt],l,mid,L,R)+query(rs[rt],mid+1,r,L,R);
}
int qkth(int l,int r,int x)
{
    if(l==r)return l;
    int res=0;
    for(auto i:tmp)res+=sum[ls[i]];
    for(auto i:tmp2)res-=sum[ls[i]];
    if(res>=x)
    {
        for(int i=0;i<tmp.size();i++)tmp[i]=ls[tmp[i]];
        for(int i=0;i<tmp2.size();i++)tmp2[i]=ls[tmp2[i]];
        return qkth(l,mid,x);
    }
    else
    {
        for(int i=0;i<tmp.size();i++)tmp[i]=rs[tmp[i]];
        for(int i=0;i<tmp2.size();i++)tmp2[i]=rs[tmp2[i]];
        return qkth(mid+1,r,x-res);
    }
}
```

```
void add(int x,int num,int z){for(int
i=x;i<=n;i+=lowbit(i))change(rt[i],0,inf,num,z);}
int qry(int x,int y)//查询1->x 中比 y 小元素个数
{int res=0;for(int i=x;i;i-=lowbit(i))res+=query(rt[i],0,inf,0,y-1);return res;}
int queryrnk(int l,int r,int x){return qry(r,x)-qry(l-1,x)+1;}
int querykth(int l,int r,int x)
{
    tmp.clear(),tmp2.clear();
    for(int i=r;i;i-=lowbit(i))tmp.pb(rt[i]);
    if(l>1)for(int i=l-1;i;i-=lowbit(i))tmp2.pb(rt[i]);
    return qkth(0,inf,x);
}
int querypre(int l,int r,int x)
{
    int z=queryrnk(l,r,x);
    if(z==1)return -2147483647;
    else return querykth(l,r,z-1);
}
int querynex(int l,int r,int x)
{
    int z=queryrnk(l,r,x+1);
    if(z==r-l+2)return 2147483647;
    else return querykth(l,r,z);
}
signed main()
{
    n=read(),m=read();
    for(int i=1;i<=n;i++)a[i]=read(),add(i,a[i],1);
    //cout<<qry(4,inf)<<endl;
    for(int i=1;i<=m;i++)
    {
        op=read();l=read(),r=read();
        if(op==3){add(l,a[l],-1),a[l]=r,add(l,a[l],1);continue;}x=read();
        if(op==1)printf("%d\n",queryrnk(l,r,x));
        if(op==2)printf("%d\n",querykth(l,r,x));
        if(op==4)printf("%d\n",querypre(l,r,x));
        if(op==5)printf("%d\n",querynex(l,r,x));
    }
    return 0;
}
```

1.8线段树分治（离线技巧 /包含了按秩合并并查集）

有一个 n 个节点的图。

在 k 时间内有 m 条边会出现后消失。

要求出每一时间段内这个图是否是二分图。

```
#include<bits/stdc++.h>
using namespace std;
#define re register
#define ls (rt<<1)
#define rs (rt<<1|1)
```

```
#define mid ((l+r)>>1)
#define fi first
#define se second
const int maxn=4e5+5,maxm=2e5+5;
vector<int>t[maxn];
stack<pair<int,int>>st;
int n,m,k,f[maxn],u[maxn],v[maxn],d[maxn],l,r;
inline int read()
{
    char ch=getchar();bool f=0;int x=0;
    for(;!isdigit(ch);ch=getchar())if(ch=='-')f=1;
    for(;isdigit(ch);ch=getchar())x=(x<<1)+(x<<3)+(ch^48);
    if(f==1)x=-x;return x;
}
int getf(int x)
{
    while(x!=f[x])x=f[x];
    return x;
}
void ins(int rt,int l,int r,int L,int R,int num)
{
    if(l>R||r<L)return;
    if(l>=L&&r<=R){t[rt].push_back(num);return;}
    ins(ls,l,mid,L,R,num),ins(rs,mid+1,r,L,R,num);
}
void merge(int x,int y)
{
    if(x==y)return;
    if(d[x]>d[y])swap(x,y);
    st.push({x,d[x]==d[y]});
    d[y]+=(d[x]==d[y]);f[x]=y;
}
void solve(int rt,int l,int r)
{
    int F=0,s=st.size();
    for(auto i:t[rt])
    {
        int x=getf(u[i]),y=getf(v[i]);
        if(x==y)
        {
            for(int j=l;j<=r;j++)puts("No");
            F=1;break;
        }merge(getf(u[i]+n),y),merge(getf(v[i]+n),x);
    }
    if(!F)
    {
        if(l==r)puts("Yes");
        else solve(ls,l,mid),solve(rs,mid+1,r);
    }
    while(st.size()>s)d[f[st.top().fi]]-
=st.top().se,f[st.top().fi]=st.top().fi,st.pop();
}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
```

```
    n=read(),m=read(),k=read();
    for(int i=1;i<=m;i++)
    {
        u[i]=read(),v[i]=read(),l=read(),r=read();
        ins(1,1,k,l+1,r,i);
    }
    for(int i=1;i<=n;i++)f[i]=i,f[n+i]=n+i;
    solve(1,1,k);
    return 0;
}
```

1.9CDQ 分治

有 n 个元素，第 i 个元素有 a_i, b_i, c_i 三个属性，设 $f(i)$ 表示满足 $a_j \leq a_i$ 且 $b_j \leq b_i$ 且 $c_j \leq c_i$ 且 $j \neq i$ 的 j 的数量。

对于 $d \in [0, n)$，求 $f(i) = d$ 的数量。

```
#include<bits/stdc++.h>
#define lowbit(x) ((x)&(-(x)))
using namespace std;
const int maxn=100015;
inline int read()
{
    char ch=getchar();bool f=0;int x=0;
    for(;!isdigit(ch);ch=getchar())if(ch=='-')f=1;
    for(;isdigit(ch);ch=getchar())x=(x<<1)+(x<<3)+(ch^48);
    if(f==1)x=-x;return x;
}
int n,m,ans[maxn],cnt,tot=0,c[maxn*2];
struct node
{
    int x,y,z,d,w;
}a[maxn],b[maxn];
bool cmp(node a,node b)
{
    if(a.x!=b.x)return a.x<b.x;if(a.y!=b.y)return a.y<b.y;return a.z<b.z;
}
void updata(int x,int y){for(;x<=m;x+=lowbit(x)) c[x]+=y;}
int query(int x){int ans=0;for(;x;x-=lowbit(x))ans+=c[x];return ans;}
void CDQ(int l,int r)
{
    int mid=l+r>>1;int ll=l,rr=mid+1,cnt=l;
    if(l==r)return;CDQ(l,mid),CDQ(mid+1,r);
    while(ll<=mid&&rr<=r)
    {
        if(a[ll].y<=a[rr].y)updata(a[ll].z,a[ll].w),b[cnt++]=a[ll++];
        else a[rr].d+=query(a[rr].z),b[cnt++]=a[rr++];
    }
    while(ll<=mid)updata(a[ll].z,a[ll].w),b[cnt++]=a[ll++];
    while(rr<=r) a[rr].d+=query(a[rr].z),b[cnt++]=a[rr++];
    for(int i=l;i<=mid;i++)updata(a[i].z,-a[i].w);for(int i=l;i<=r;i++)a[i]=b[i];
}
```

```
int main()
{
    n=read();m=read();
    for(int i=1;i<=n;i++)a[i].x=read(),a[i].y=read(),a[i].z=read(),a[i].w=1;
    sort(a+1,a+n+1,cmp);tot=1;
    for(int i=2;i<=n;i++)
        if(a[i].x==a[tot].x&&a[i].y==a[tot].y&&a[i].z==a[tot].z)a[tot].w++;
        else a[++tot]=a[i];CDQ(1,tot);
    for(int i=1;i<=tot;i++)ans[a[i].d+a[i].w-1]+=a[i].w;
    for(int i=0;i<n;i++) printf("%d\n",ans[i]);
    return 0;
}
```

1.10莫队

1.10.1树上莫队

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=1e5+5,M=34005,inf=1e8;
inline int read(){
    char ch=getchar();bool f=0;int x=0;
    for(;!isdigit(ch);ch=getchar())if(ch=='-')f=1;
    for(;isdigit(ch);ch=getchar())x=(x<<1)+(x<<3)+(ch^48);
    if(f==1)x=-x;return x;
}
inline void print(int x){
    static int a[55];int top=0;
    if(x<0) putchar('-'),x=-x;
    do{a[top++]=x%10,x/=10;}while(x);
    while(top) putchar(a[--top]+48);
}
int s[maxn],dfn[maxn],f[maxn][22],d[maxn],vis[maxn],u[maxn],a[maxn],id[maxn];
int l,r,n,m,x,y,k,tot=0,cnt=0,ls[maxn*202],rs[maxn*202],sum[maxn*202],rt[maxn];
vector<int>b,e[maxn];
void upd(int &rt,int l,int r,int pos,int num){
    if(l>pos||r<pos)return;if(!rt)rt=++tot;
    sum[rt]+=num;if(l==r)return;
    upd(ls[rt],l,mid,pos,num);upd(rs[rt],mid+1,r,pos,num);
}
void add(int x,int pos,int num){for(int i=x;i<=n;i+=lowbit(i))
{upd(rt[i],0,inf,pos,num);}}
int Lca(int x,int y){
    if(d[x]<d[y])swap(x,y);
    for(int i=20;i>=0;i--)if(d[f[x][i]]>=d[y])x=f[x][i];
    if(x==y)return x;
    for(int i=20;i>=0;i--)if(f[x][i]!=f[y][i])x=f[x][i],y=f[y][i];
    return f[x][0];
}
void dfs(int x,int fa){
    s[x]=1;dfn[x]=++cnt;id[cnt]=x;d[x]=d[fa]+1;f[x][0]=fa;
    for(int i=1;i<=20;i++)f[x][i]=f[f[x][i-1]][i-1];
    for(auto i:e[x])if(i^fa){
        dfs(i,x),s[x]+=s[i];
    }
```

```
}
void solve(int x,int y,int k){
    int g=Lca(x,y),p=f[g][0];k=d[x]+d[y]-d[g]*2+1-k+1;
    x=dfn[x],y=dfn[y],g=dfn[g],p=dfn[p];
    if(k<=0){puts("invalid request!");return;}b.clear();
    for(int i=x;i;i-=lowbit(i))vis[i]=1,b.pb(i);
    for(int i=y;i;i-=lowbit(i))if(!vis[i])vis[i]=1,b.pb(i);
    for(int i=g;i;i-=lowbit(i))if(!vis[i])vis[i]=1,b.pb(i);
    for(int i=p;i;i-=lowbit(i))if(!vis[i])vis[i]=1,b.pb(i);
    for(auto i:b)u[i]=rt[i];
    int l=0,r=inf;//cout<<l<<" "<<r<<endl;return 0;
    while(l<=r){
        if(l==r){printf("%d\n",l);break;}
        int num=0;
        for(int i=x;i;i-=lowbit(i))num+=sum[ls[u[i]]];
        for(int i=y;i;i-=lowbit(i))num+=sum[ls[u[i]]];
        for(int i=g;i;i-=lowbit(i))num-=sum[ls[u[i]]];
        for(int i=p;i;i-=lowbit(i))num-=sum[ls[u[i]]];
        if(num>=k){for(auto i:b)u[i]=ls[u[i]];r=mid;}
        else {for(auto i:b)u[i]=rs[u[i]];k-=num;l=mid+1;}
    }
    for(auto i:b)vis[i]=0;
}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    n=read(),m=read();
    for(int i=1;i<=n;i++)a[i]=read();
    for(int i=1;i<n;i++)x=read(),y=read(),e[x].pb(y),e[y].pb(x);
    dfs(1,0);
    for(int i=1;i<=n;i++)add(dfn[i],a[i],1),add(dfn[i]+s[i],a[i],-1);
    for(int i=1;i<=m;i++){
        k=read(),x=read(),y=read();
        if(k==0)add(dfn[x],a[x],-1),add(dfn[x]+s[x],a[x],1),
        a[x]=y,add(dfn[x],a[x],1),add(dfn[x]+s[x],a[x],-1);
        else solve(x,y,k);
    }
    return 0;
}
```

1.10.2带修莫队

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=1050005;
struct node
{
    int l,r,id,pre;
}b[maxn];
int
n,m,a[maxn],t,t2,id[maxn],s[maxn],ans[maxn],sum=0,k,tot=0,c[maxn],c2[maxn],f[maxn
],cnt=0,l,r,p;
char ch;
bool cmp(node a,node b)
```

```
{
    if(id[a.l]!=id[b.l])return a.l<b.l;
    if(id[a.r]!=id[b.r])return a.r<b.r;
    return a.pre<b.pre;
}
void add(int x){s[x]++;if(s[x]==1)sum++;}
void cut(int x){s[x]--;if(s[x]==0)sum--;}
void change(int x)
{
    if(c[x]<=r&&c[x]>=l)
    {
        s[a[c[x]]]--;if(!s[a[c[x]]])sum--;
        if(!s[c2[x]])sum++;s[c2[x]]++;
    }
    swap(c2[x],a[c[x]]);
}
signed main()
{
    n=read(),m=read();
    for(int i=1;i<=n;i++)a[i]=read();
    for(int i=1;i<=m;i++)
    {
        while(ch!='Q'&&ch!='R')ch=getchar();

if(ch=='Q')++cnt,b[cnt].l=read(),b[cnt].r=read(),b[cnt].id=cnt,b[cnt].pre=tot;
        else c[++tot]=read(),c2[tot]=read();
        ch=0;
    }t=ceil(exp((log(n)+log(cnt))/3));
    for(int i=1;i<=n;i++)id[i]=(i-1)/t+1;
    sort(b+1,b+cnt+1,cmp);l=0,r=0,p=0,sum=0;
    for(int i=1;i<=cnt;i++)
    {
        while(l<b[i].l)cut(a[l]),l++;
        while(l>b[i].l)l--,add(a[l]);
        while(r<b[i].r)r++,add(a[r]);
        while(r>b[i].r)cut(a[r]),r--;
        while(p<b[i].pre)p++,change(p);
        while(p>b[i].pre)change(p),p--;
        ans[b[i].id]=sum;
    }
    for(int i=1;i<=cnt;i++)printf("%d\n",ans[i]);
    return 0;
}
```

1.10.3回滚莫队

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=400005;
struct node{
    int l,r,id;
}b[maxn];
int
m,n,x,y,a[maxn],t,id[maxn],ans[maxn],sum=0,k,l,r,num=0,pos[maxn],f[maxn],c[maxn],
cl[maxn],tot=0;
```

```
bool cmp(node a,node b){
    if(id[a.l]!=id[b.l])return id[a.l]<id[b.l];
    else return a.r<b.r;
}
int main(){
    n=read();t=sqrt(n);
    for(int i=1;i<=n;i++)c[i]=a[i]=read(),id[i]=(i-
1)/t+1;m=read();sort(c+1,c+n+1);
    int un=unique(c+1,c+1+n)-c-1;
    for(int i=1;i<=n;i++) a[i]=lower_bound(c+1,c+1+un,a[i])-c;
    for(int i=1;i<=m;i++)b[i].l=read(),b[i].r=read(),b[i].id=i;
    sort(b+1,b+m+1,cmp);
    for(int i=1;i<=m;i++){
        if(id[b[i].l]!=id[b[i-1].l]){
            memset(pos,0,sizeof pos),memset(f,0,sizeof f);
            k=min(n,id[b[i].l]*t),r=k,num=0,tot=0;
        }
        if(id[b[i].r]==id[b[i].l]){
            sum=0;
            for(int j=b[i].l;j<=b[i].r;j++)
                if(!pos[a[j]])pos[a[j]]=j;
                else sum=max(j-pos[a[j]],sum);
            for(int j=b[i].l;j<=b[i].r;j++)
                pos[a[j]]=0;ans[b[i].id]=sum;
        }
        else {
            while(r<b[i].r){
                r++;if(!f[a[r]])f[a[r]]=r,cl[++tot]=a[r];
                pos[a[r]]=r;num=max(num,pos[a[r]]-f[a[r]]);
            }
            sum=num;l=k+1;
            while(l>b[i].l){
                l--,sum=max(sum,pos[a[l]]-l);
                ans[b[i].id]=sum;if(!pos[a[l]])pos[a[l]]=l;
            }
            while(l!=k+1){if(pos[a[l]]==l)pos[a[l]]=0;l++;}
        }
    }
    for(int i=1;i<=m;i++)printf("%d\n",ans[i]);
    return 0;
}
```

1.11 点分树（点分治的在线）

```
#include<bits/stdc++.h>
using namespace std;
int n,m,Min,cnt,inf=1e9,x,y,sum,rt,ans,opt;
int val[maxn],h[maxn],sz[maxn],fa[maxn];
int top[maxn],f[maxn],son[maxn],siz[maxn],d[maxn];
vector<int>e[maxn],c[2][maxn];bool vis[maxn];
void dfs1(int x,int fa){
    f[x]=fa,d[x]=d[fa]+1;siz[x]=1;
    int Max=0;
```

```
    for(auto i:e[x])
        if(i^fa){
            dfs1(i,x),siz[x]+=siz[i];
            if(siz[i]>Max)Max=siz[i],son[x]=i;
        }
}
void dfs2(int x,int tf){
    top[x]=tf;if(son[x])dfs2(son[x],tf);
    for(auto i:e[x])
        if((i^f[x])&&(i^son[x]))dfs2(i,i);
}
int Lca(int x,int y){
    while(top[x]!=top[y]){
        if(d[top[x]]<d[top[y]])swap(x,y);
        x=f[top[x]];
    }
    if(d[x]>d[y])swap(x,y);return x;
}
int getdis(int u,int v){int lca=Lca(u,v);return d[u]+d[v]-2*d[lca];}
void add(int u,int opt,int x,int val)
{for(int i=x+1;i<=sz[u];i+=lowbit(i))c[opt][u][i]+=val;}
int qry(int u,int opt,int x){
    int res=0;for(int i=min(x+1,sz[u]);i;i-=lowbit(i))res+=c[opt][u][i];return
res;
}
void findrt(int x,int f){
    sz[x]=1;int res=0;
    for(auto v:e[x])if(v!=f&&!vis[v])
        findrt(v,x),sz[x]+=sz[v],res=max(res,sz[v]);
    res=max(res,sum-sz[x]);
    if(res<Min)Min=res,rt=x;
}
void dfs(int x){
    vis[x]=1;sz[x]=sum+1;
    c[0][x].resize(sz[x]+1),c[1][x].resize(sz[x]+1);
    for(auto v:e[x])
        if(!vis[v]){
            sum=sz[v];rt=0;Min=inf;findrt(v,0);
            fa[rt]=x,dfs(rt);
        }
}
void modify(int x,int num)
{
    for(int i=x;i;i=fa[i])add(i,0,getdis(x,i),num);
    for(int i=x;fa[i];i=fa[i])add(i,1,getdis(x,fa[i]),num);
}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    n=read();m=read();
    for(int i=1;i<=n;i++)val[i]=read();
    for(int i=1;i<n;i++)x=read(),y=read(),e[x].push_back(y),e[y].push_back(x);
    dfs1(1,0);dfs2(1,1);
    sum=n,Min=inf;
    findrt(1,0);dfs(rt);
```

```
    for(int i=1;i<=n;i++)modify(i,val[i]);
    while(m--)
    {
        opt=read(),x=read(),y=read();x=x^ans,y=y^ans;
        if(opt==0)
        {
            ans=0,ans+=qry(x,0,y);
            for(int i=x;fa[i];i=fa[i])
            {
                int dis=getdis(x,fa[i]);
                if(y>=dis)ans+=qry(fa[i],0,y-dis)-qry(i,1,y-dis);
            }
            print(ans);puts("");
        }
        else modify(x,y-val[x]),val[x]=y;
    }
    return 0;
}
```

1.12 dsu on the tree /启发式合并

用重儿子性质，暴力处理清儿子。

```
#include<bits/stdc++.h>
using namespace std;
#define int long long
const int maxn=1e5+5;
inline int read()
{
    char ch=getchar();int x=0;bool f=0;
    for(;!isdigit(ch);ch=getchar())if(ch=='-')f=1;
    for(;isdigit(ch);ch=getchar())x=(x<<1)+(x<<3)+(ch^48);
    if(f==1)x=-x;return x;
}
struct node
{
    int v,nex;
}e[maxn*2];
int
n,m,a[maxn],b[maxn],head[maxn],s[maxn],son[maxn],x,y,ans[maxn],sum,Max=0,cnt,c[ma
xn];
void add(int x,int y){e[++cnt].v=y;e[cnt].nex=head[x];head[x]=cnt;}
void dfs(int x,int fa)
{
    int p=0;s[x]=1;
    for(int i=head[x];i;i=e[i].nex)
    {
        int v=e[i].v;if(v==fa)continue;
        dfs(v,x);s[x]+=s[v];if(s[v]>p)p=s[v],son[x]=v;
    }
}
void del(int x,int fa)
{
    sum=0;c[a[x]]--;
    for(int i=head[x];i;i=e[i].nex)
```

```
    {
        int v=e[i].v;if(v==fa)continue;
        del(v,x);
    }
}
void dfs2(int x,int fa)
{
    c[a[x]]++;if(c[a[x]]==1)sum++;
    for(int i=head[x];i;i=e[i].nex)
    {
        int v=e[i].v;if(v==fa)continue;
        dfs2(v,x);
    }
}
void dfs1(int x,int fa,int num)
{
    for(int i=head[x];i;i=e[i].nex)
    {
        int v=e[i].v;if(v==fa||son[x]==v)continue;
        dfs1(v,x,0);
    }
    if(son[x])dfs1(son[x],x,1);
    c[a[x]]++;if(c[a[x]]==1)sum++;
    for(int i=head[x];i;i=e[i].nex)
    {
        int v=e[i].v;if(v==fa||son[x]==v)continue;
        dfs2(v,x);
    }ans[x]=sum;
    if(num==0)del(x,fa),Max=0;

}
signed main()
{
    n=read();
    for(int i=1;i<n;i++)x=read(),y=read(),add(x,y),add(y,x);
    for(int i=1;i<=n;i++)a[i]=read();
    dfs(1,0);dfs1(1,0,1);m=read();
    for(int i=1;i<=m;i++)x=read(),printf("%lld\n",ans[x]);
    return 0;
}
```

1.13 笛卡尔树

给定一个 $1 \sim n$ 的排列 p，构建其笛卡尔树。

即构建一棵二叉树，满足：

1. 每个节点的编号满足二叉搜索树的性质。
2. 节点 i 的权值为 p_i，每个节点的权值满足小根堆的性质。

```
int st[maxn],n,m,ls[maxn],rs[maxn],top=0,a[maxn];
long long ans1=0,ans2=0;
int main()
{
    n=read();
```

```
    for(int i=1;i<=n;i++)
    {
        a[i]=read();int pos=top;
        while(a[i]<a[st[top]]&&top>0)top--;
        if(pos>top)ls[i]=st[top+1];
        if(top!=0)rs[st[top]]=i;st[++top]=i;
    }
    for(int i=1;i<=n;i++)ans1=ans1^(1ll*i*(ls[i]+1)),ans2=ans2^(1ll*i*(rs[i]+1));
    cout<<ans1<<" "<<ans2<<endl;
    return 0;
}
```

1.14 RMQ

```
int n,m,lg[maxn],st[maxn][22],l,r;
signed main()
{
    n=read(),m=read();lg[0]=-1;
    for(int i=1;i<=n;i++)lg[i]=lg[i>>1]+1,st[i][0]=read();
    for(int j=1;j<=21;j++)
        for(int i=1;i+(1<<j)-1<=n;i++)
            st[i][j]=max(st[i][j-1],st[i+(1<<(j-1))][j-1]);
    for(int i=1;i<=m;i++)
    {
        l=read(),r=read();
        int z=lg[r-l+1];
        print(max(st[l][z],st[r-(1<<z)+1][z]));
        puts("");
    }
    return 0;
}
```

1.15 李超树

```
double k[maxn],b[maxn],Max;
int n,m,tag[maxn*4],v[maxn*4];
int x,y,ans,opt,las,x2,y2,tot=0;
double get(int i,int x){return k[i]*x+b[i];}
void add(int rt,int l,int r,int L,int R,int x){
    if(l>R||r<L)return;
    if(l>=L&&r<=R){
        if(!tag[rt]){tag[rt]=x;return;}
        if(get(x,mid)-get(tag[rt],mid)>eps)swap(tag[rt],x);
        if(l==r){return;}
        if(k[tag[rt]]>k[x]){add(ls,l,mid,L,R,x);}
        else add(rs,mid+1,r,L,R,x);
        return;
    }add(ls,l,mid,L,R,x);add(rs,mid+1,r,L,R,x);
}
pair<double,int>query(int rt,int l,int r,int x){
    if(l>x||r<x)return mp(0,0);
    if(l==r)return mp(get(tag[rt],x),-tag[rt]);
    double sum=0;int id;
    pair<double,int>u=max(query(ls,l,mid,x),query(rs,mid+1,r,x));
```

```
    if(tag[rt])sum=get(tag[rt],x);
    if(fabs(u.fi-sum)<eps){id=min(tag[rt],-u.se);return mp(sum,-id);}
    return max(mp(sum,-tag[rt]),u);
}
signed main(){
    //freopen("1.in","r",stdin);
    //freopen("1.out","w",stdout);
    n=read();
    for(int i=1;i<=n;i++){
        opt=read();
        if(opt==0){
            x=read(),x=(x+las-1)%39989+1;
            pi ans=query(1,1,N,x);
            las=-ans.se;printf("%d\n",las);
        }
        if(opt==1){
            x=read(),y=read(),x2=read(),y2=read();
            x=(x+las-1)%39989+1,y=(y+las-1)%mod+1,x2=(x2+las-1)%39989+1,y2=
(y2+las-1)%mod+1;
            if(x>x2)swap(x,x2),swap(y,y2);
            if(x==x2)k[++tot]=0,b[tot]=y2;
            else k[++tot]=1.0*(y2-y)/(x2-x),b[tot]=1.0*y-1.0*k[tot]*x;
            add(1,1,N,x,x2,tot);
        }
    }
    return 0;
}
```

2 字符串

2.1AC自动机

（kmp套字典树）

给你一个文本串 *S* 和 *n* 个模式串 *T*1~*n*，请你分别求出每个模式串 Ti 在 S 中出现的次数。

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=2e6+5;
int n,m,tot=1,pos[maxn],in[maxn];
int ch[maxn][26],fail[maxn],sum[maxn],ans[maxn];
char a[maxn],b[maxn];
void ins(int id){
    int len=strlen(a+1);
    int now=1;
    for(int i=1;i<=len;i++){
        if(!ch[now][a[i]-'a']) ch[now][a[i]-'a']=++tot;
        now=ch[now][a[i]-'a'];
    }pos[id]=now;
}
void getfail(){
    for(int i=0;i<26;i++)ch[0][i]=1;
    queue<int>q;q.push(1);
    while(!q.empty()){
        int x=q.front();q.pop();
```

```
        for(int i=0;i<26;i++)
            if(ch[x][i])
                fail[ch[x][i]]=ch[fail[x]][i],
                q.push(ch[x][i]),in[fail[ch[x][i]]]++;
            else ch[x][i]=ch[fail[x]][i];
    }
}
void topsort(){
    queue<int>q;
    for(int i=1;i<=tot;i++)if(in[i]==0)q.push(i);
    while(!q.empty()){
        int x=q.front();q.pop();
        in[fail[x]]--;sum[fail[x]]+=sum[x];
        if(in[fail[x]]==0)q.push(fail[x]);
    }
}
signed main(){
    n=read();
    for(int i=1;i<=n;i++){
        scanf("%s",a+1);ins(i);
    }
    scanf("%s",b+1);getfail();
    int len=strlen(b+1),now=1;
    for(int i=1;i<=len;i++){
        now=ch[now][b[i]-'a'];
        sum[now]++;
    }topsort();
    for(int i=1;i<=n;i++)
        printf("%d\n",sum[pos[i]]);
    return 0;
}
```

2.2后缀自动机

请你求出 *S* 的所有出现次数不为 1 的子串的出现次数乘上该子串长度的最大值。

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=2e6+5,M=34005;
char s[maxn];
int m,ch[maxn][26],fa[maxn],len[maxn],p,t=1,tot=1,sz[maxn];
long long ans=0;vector<int>e[maxn];
void ins(int x){
    p=t;t=++tot;len[t]=len[p]+1;sz[t]=1;
    for(;p&&!ch[p][x];p=fa[p])ch[p][x]=t;
    if(!p){fa[t]=1;return;}int q=ch[p][x];
    if(len[q]==len[p]+1){fa[t]=q;return;}
    ++tot;len[tot]=len[p]+1;fa[tot]=fa[q];fa[q]=fa[t]=tot;
    memcpy(ch[tot],ch[q],sizeof ch[q]);
    for(int i=p;ch[i][x]==q;i=fa[i])ch[i][x]=tot;
}
void build(){for(int i=2;i<=tot;i++)e[fa[i]].pb(i);}
void dfs(int x){
    for(auto i:e[x]){
        dfs(i);sz[x]+=sz[i];
```

```
    }if(sz[x]!=1)ans=max(ans,1ll*sz[x]*len[x]);
}
signed main()
{
    //freopen("1.in","r",stdin);
    //freopen(".out","w",stdout);
    scanf("%s",s+1);m=strlen(s+1);
    for(int i=1;i<=m;i++)ins(s[i]-'a');
    build();dfs(1);cout<<ans;
    return 0;
}
```

2.2.1 后缀自动机加数据结构

给两个串，问B的字串有多少个不是A的字串

```
#include<bits/stdc++.h>
using namespace std;
#define pb push_back
#define mid (l+r>>1)
#define ll long long
const int maxn=1e6,M=34005;
int n,m,ls[maxn*32],rs[maxn*32],rt[maxn],tot=0,l,r,g[maxn];
char a[maxn],b[maxn];
void add(int &rt,int l,int r,int pos){
    if(l>pos||r<pos)return;
    rt=++tot;if(l==r)return;
    add(ls[rt],l,mid,pos),add(rs[rt],mid+1,r,pos);
}
int merge(int x,int y,int l,int r){
    if(!x||!y)return x+y;
    int rt=++tot;if(l==r)return rt;
    ls[rt]=merge(ls[x],ls[y],l,mid);
    rs[rt]=merge(rs[x],rs[y],mid+1,r);return rt;
}
bool query(int rt,int l,int r,int L,int R){
    if(l>R||r<L||!rt)return 0;
    if(l>=L&&r<=R)return 1;
    return query(ls[rt],l,mid,L,R)|query(rs[rt],mid+1,r,L,R);
}
struct SAM{
    int len[maxn],tag[maxn],ch[maxn][26],fa[maxn],las=1,tot=1;
    vector<int>e[maxn];
    void clear(){
        for(int i=1;i<=tot;i++)
            memset(ch[i],0,sizeof ch[i]),fa[i]=0,tag[i]=0,len[i]=0;
        tot=las=1;
    }
    void ins(int x,int pos){
        int p=las,t=++tot;las=tot;len[t]=len[p]+1;add(rt[t],1,n,pos);
        for(;!ch[p][x];p=fa[p])ch[p][x]=t;
        if(!p){fa[t]=1;return;}int q=ch[p][x];
        if(len[p]+1==len[q]){fa[t]=q;return;}
        len[++tot]=len[p]+1;fa[tot]=fa[q];fa[q]=fa[t]=tot;
        memcpy(ch[tot],ch[q],sizeof ch[q]);add(rt[t],1,n,pos);
```

```
        for(;ch[p][x]==q;p=fa[p])ch[p][x]=tot;
    }
    void ins2(int x){
        int p=las,t=++tot;las=tot;len[t]=len[p]+1;tag[t]=len[t];
        for(;!ch[p][x];p=fa[p])ch[p][x]=t;
        if(!p){fa[t]=1;return;}int q=ch[p][x];
        if(len[p]+1==len[q]){fa[t]=q;return;}
        len[++tot]=len[p]+1;fa[tot]=fa[q];fa[q]=fa[t]=tot;
        memcpy(ch[tot],ch[q],sizeof ch[q]);tag[tot]=tag[q];
        for(;ch[p][x]==q;p=fa[p])ch[p][x]=tot;
    }
    void dfs(int x){
        for(auto i:e[x])
            dfs(i),rt[x]=merge(rt[x],rt[i],1,n);
    }
    void work(){
        for(int i=2;i<=tot;i++)e[fa[i]].pb(i);dfs(1);
    }
}S,T;
void solve(int l,int r,int m){
    int p=1,lx=0;
    ll ans=0;
    for(int i=1;i<=m;i++){
        int x=b[i]-'a';T.ins2(x);
        while(true){
            if(S.ch[p][x]&&query(rt[S.ch[p][x]],1,n,lx+l,r))
                {p=S.ch[p][x];lx++;break;}
            if(!lx)break;lx--;
            if(S.len[S.fa[p]]==lx)p=S.fa[p];
        }g[i]=lx;
    }
    for(int i=2;i<=T.tot;i++)
        ans+=max(0,T.len[i]-max(T.len[T.fa[i]],g[T.tag[i]]));
    for(int i=1;i<=m;i++)g[i]=0;
    printf("%lld\n",ans);
}
signed main(){
    scanf("%s",a+1);n=strlen(a+1);
    for(int i=1;i<=n;i++)S.ins(a[i]-'a',i);m=read();S.work();
    for(int i=1;i<=m;i++){
        T.clear();scanf("%s",b+1);
        l=read(),r=read();solve(l,r,strlen(b+1));
    }
    return 0;
}
```

一些经典操作

求最长公共子串（多个串）

```
int n,m,c[maxn],sa[maxn];
char a[maxn];
struct SAM{
    int len[maxn],fa[maxn],tot=1,las=1,ch[maxn][26],mx[maxn],mn[maxn];
    void ins(int x){
```

```
        int p=las,t=++tot;las=tot;len[t]=len[p]+1;mn[t]=len[t];
        for(;!ch[p][x];p=fa[p])ch[p][x]=t;
        if(!p){fa[t]=1;return;}int q=ch[p][x];
        if(len[q]==len[p]+1){fa[t]=q;return;}
        len[++tot]=len[p]+1;fa[tot]=fa[q];fa[q]=fa[t]=tot;mn[tot]=len[tot];
        memcpy(ch[tot],ch[q],sizeof ch[q]);
        for(;ch[p][x]==q;p=fa[p])ch[p][x]=tot;
    }
    void work(){

        for(int i=1;i<=tot;i++) ++c[len[i]];
        for(int i=1;i<=tot;i++) c[i]+=c[i-1];
        for(int i=1;i<=tot;i++) sa[c[len[i]]--]=i;
    }
    void solve(int n){
        int x=1,l=0;for(int i=2;i<=tot;i++)mx[i]=0;
        for(int i=1;i<=n;i++){
            while(x&&!ch[x][a[i]-'a'])x=fa[x],l=len[x];
            if(!x)x=1,l=0;
            if(ch[x][a[i]-'a'])x=ch[x][a[i]-'a'],l++;mx[x]=max(mx[x],l);
        }
        for(int i=tot;i>=1;i--){
            int x=sa[i];
            mx[fa[x]]=max(mx[x],mx[fa[x]]);
            mn[x]=min(mn[x],mx[x]);
        }
    }
}sam;
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    while(~scanf("%s",a+1)){
        n++;m=strlen(a+1);
        if(n==1){for(int i=1;i<=m;i++)sam.ins(a[i]-'a');sam.work();}
        else sam.solve(m);
    }int ans=0;
    for(int i=1;i<=sam.tot;i++)ans=max(ans,sam.mn[i]);
    cout<<ans;
    return 0;
}
```

不同子串个数

```
char s[maxn];
int len;
long long ans=0;
struct Sam
{
    int ch[maxn][27],len[maxn],fa[maxn];
    long long f[maxn];
    int t=1,tot=1;vector<int>e[maxn];
    void ins(int x)
    {
        int p=t;t=++tot;len[t]=len[p]+1;
```

```
        for(;p&&!ch[p][x];p=fa[p])ch[p][x]=t;
        if(!p){fa[t]=1;return;}int q=ch[p][x];
        if(len[q]==len[p]+1){fa[t]=q;return;}
        len[++tot]=len[p]+1;fa[tot]=fa[q];fa[q]=fa[t]=tot;
        memcpy(ch[tot],ch[q],sizeof(ch[tot]));
        for(int i=p;ch[i][x]==q;i=fa[i])ch[i][x]=tot;
    }
    long long dfs(int x)
    {
        //cout<<x<<endl;
        if(f[x])return f[x];
        for(int i=0;i<26;i++)if(ch[x][i])f[x]+=dfs(ch[x][i])+1;
        return f[x];

    }
}sam;
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    len=read();scanf("%s",s+1);
    for(int i=1;i<=len;i++)sam.ins(s[i]-'a');
    cout<<sam.dfs(1);
    return 0;
}
```

字典序第K大

```
int n,m,x,y;
char s[maxn];
struct Sam
{
    int tot=1,t=1,len[maxn],sz[maxn],fa[maxn],f[maxn];
    int ch[maxn][26];vector<int>e[maxn];
    void ins(int x)
    {
        int p=t;t=++tot;len[tot]=len[p]+1;sz[t]=1;
        for(;p&&!ch[p][x];p=fa[p])ch[p][x]=t;
        if(!p){fa[t]=1;return;}int q=ch[p][x];
        if(len[p]+1==len[q]){fa[t]=q;return;}
        len[++tot]=len[p]+1;fa[tot]=fa[q];fa[q]=fa[t]=tot;
        memcpy(ch[tot],ch[q],sizeof ch[tot]);
        for(int i=p;ch[i][x]==q;i=fa[i])ch[i][x]=tot;
    }
    void build(){for(int i=2;i<=tot;i++)e[fa[i]].pb(i);}
    void dfs1(int x){for(auto i:e[x])dfs1(i),sz[x]+=sz[i];}
    int dfs2(int x){if(f[x])return f[x];sz[x]=1,f[x]=1;for(int
i=0;i<26;i++)if(ch[x][i])f[x]+=dfs2(ch[x][i]);return f[x];}
    int dfs3(int x){if(f[x])return f[x];f[x]=sz[x];for(int i=0;i<26;i++)if(ch[x]
[i])f[x]+=dfs3(ch[x][i]);return f[x];}
    void query(int x,int k)
    {
        if(k==0)return;
        for(int i=0;i<26;i++)
        {
```

```
            if(f[ch[x][i]]>=k){putchar(i+'a'),query(ch[x][i],k-sz[ch[x]
[i]]);return;}
            else k-=f[ch[x][i]];
        }
    }

}sam;
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    scanf("%s",s+1);n=strlen(s+1);
    for(int i=1;i<=n;i++)sam.ins(s[i]-'a');
    x=read(),y=read();
    if(x==0)sam.dfs2(1),sam.sz[1]=1;
    else sam.build(),sam.dfs1(1),sam.dfs3(1);
    if(sam.f[1]-sam.sz[1]<y)puts("-1");
    else sam.query(1,y);
    return 0;
}
```

一些性质：

1. 反串的 SAM 的parent树就是后缀树。(直接建应该是前缀树)
2. 设 $lcs(i,j)$ 为前缀 i,j 的最长公共后缀长度,其等于$parent$ 树上 `LCA` 的 len 值。
3. 广义SAM， 每次插串把 las 变为 1 。
4. parent 树上,父节点的 endpos 恰恰是子节点的 endpos 求并，可用线段树合并维护。
5.

2.3 KMP

```
int n,m,nex[maxn],lena,lenb,st[maxn],tot=0;
char a[maxn],b[maxn];
signed main()
{
    //freopen("P3375_11.in","r",stdin);
//  freopen(".out","w",stdout);
    scanf("%s",a+1);lena=strlen(a+1);
    scanf("%s",b+1);lenb=strlen(b+1);
    int j=0;nex[1]=0;
    for(int i=2;i<=lenb;i++)
    {
        while(j&&b[j+1]!=b[i])j=nex[j];
        if(b[j+1]==b[i])j++;nex[i]=j;
    }j=0;
    for(int i=1;i<=lena;i++)
    {
        while(j&&b[j+1]!=a[i])j=nex[j];
        if(b[j+1]==a[i])j++;
        if(j==lenb)printf("%d\n",i-lenb+1),j=nex[j];
    }
```

```
    for(int i=1;i<=lenb;i++)printf("%d ",nex[i]);
    return 0;
}
```

2.4 扩展kmp

```
int n,m,z[maxn],lena,lenb,p[maxn],tot=0,ans1,ans2;
char a[maxn],b[maxn];
void getz()
{
    z[1]=lenb;int id=0,r=0;
    for(int i=2;i<=lenb;i++)
    {
        if(i<=r)z[i]=min(z[i-id+1],r-i+1);
        while(i+z[i]<=lenb&&b[i+z[i]]==b[z[i]+1])++z[i];
        if(i+z[i]-1>r)id=i,r=i+z[i]-1;
    }
}
void exkmp()
{
    for(int i=1,id=0,r=0;i<=lena;i++)
    {
        if(i<=r)p[i]=min(z[i-id+1],r-i+1);
        while(i+p[i]<=lena&&a[i+p[i]]==b[p[i]+1])++p[i];
        if(i+p[i]-1>r)id=i,r=i+p[i]-1;
    }
}
signed main()
{
    scanf("%s",a+1);lena=strlen(a+1);
    scanf("%s",b+1);lenb=strlen(b+1);
    getz();exkmp();
    for(int i=1;i<=lenb;i++)ans1=ans1^(i*(z[i]+1));//,cout<<z[i]<<" ";cout<<endl;
    for(int i=1;i<=lena;i++)ans2=ans2^(i*(p[i]+1));//cout<<s[i]<<" ";
    cout<<ans1<<endl<<ans2<<endl;
    return 0;
}
```

3 图论

3.1 tarjan

3.1.1 强连通分量

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=2e5+5;
struct edge{int u,v,nex;}e[10000005];
int
n,m,head[maxn],st[maxn],top=0,dfn[maxn],idx=0,low[maxn],c[maxn],num[maxn],in[maxn
],out[maxn],tot=0,cnt=0,x,y,ans1=0,ans;
void tarjan(int x)
```

```
{
    dfn[x]=low[x]=++idx;st[++top]=x;
    for(int i=head[x];i;i=e[i].nex)
    {
        int v=e[i].v;
        if(!dfn[v])tarjan(v),low[x]=min(low[x],low[v]);
        else if(!c[v])low[x]=min(low[x],dfn[v]);
    }
    if(dfn[x]==low[x]){
        tot++;c[x]=tot;num[tot]++;
        while(st[top]!=x&&top>0)c[st[top]]=tot,top--,num[tot]++;
        top--;
    }
}
void add(int x,int y){e[++cnt].v=y,e[cnt].u=x;e[cnt].nex=head[x],head[x]=cnt;}
int main()
{

    n=read();
    for(int i=1;i<=n;i++){x=read();while(x!=0)add(i,x),x=read();}
    for(int i=1;i<=n;i++)if(!dfn[i])tarjan(i);
    if(tot==1)cout<<"1\n0",exit(0);
    for(int
i=1;i<=cnt;i++)if(c[e[i].u]!=c[e[i].v])in[c[e[i].v]]++,out[c[e[i].u]]++;
    for(int i=1;i<=tot;i++)
    {
        if(in[i]==0)ans++;
        if(out[i]==0)ans1++;
    }
    return 0;
}
```

3.1.2缩点

```
void tarjan(int x)
{
    dfn[x]=low[x]=++tim;st[++top]=x;vis[x]=1;
    for(int i=head[x];i;i=e[i].nex){
        if(!dfn[e[i].v])tarjan(e[i].v),low[x]=min(low[x],low[e[i].v]);
        else if(vis[e[i].v])low[x]=min(low[x],dfn[e[i].v]);
    }
    if(dfn[x]==low[x]){
        ++tot;
        while(st[top]!=x)
            id[st[top]]=tot,vv[tot]+=v[st[top]],
            vis[st[top]]=0,top--;
        id[st[top]]=tot,vv[tot]+=v[st[top]],vis[st[top]]=0,top--;
    }
}
```

3.1.3 割点（删去图不连通的点）

```
int n,m,dfn[maxn],low[maxn],vis[maxn],tot=0,rt,c=0,x,y,res=0;vector<int>e[maxn];
void tarjan(int x){
    dfn[x]=low[x]=++tot;
    for(auto i:e[x]){
        if(!dfn[i]){
            tarjan(i);low[x]=min(low[i],low[x]);
            if(x!=rt&&low[i]>=dfn[x])vis[x]=1;
            else if(x==rt)c++;
        }low[x]=min(low[x],dfn[i]);
    }
    if(x==rt&&c>=2)vis[x]=1;
}
signed main()
{
    n=read(),m=read();
    for(int i=1;i<=m;i++)x=read(),y=read(),e[x].pb(y),e[y].pb(x);
    for(int i=1;i<=n;i++)if(!dfn[i])rt=i,c=0,tarjan(i);
    for(int i=1;i<=n;i++)res+=vis[i];cout<<res<<endl;
    for(int i=1;i<=n;i++)if(vis[i])printf("%d ",i);
    return 0;
}
```

3.1.4边双（无割边）

```
int n,m,x,y,dfn[maxn],low[maxn],cnt=0,idx=0,st[maxn],top=0,tot=0,vis[maxn];
vector<pi>e[maxn];vector<int>p[maxn];
void tarjan(int x,int fa){
    dfn[x]=low[x]=++idx;
    st[++top]=x;vis[x]=1;
    for(auto i:e[x])if(i.se!=fa){
        if(!dfn[i.fi]){
            tarjan(i.fi,i.se);low[x]=min(low[x],low[i.fi]);
        }
        else low[x]=min(low[x],dfn[i.fi]);
    }
    if(dfn[x]==low[x]){
        ++tot;
        while(st[top]!=x)p[tot].pb(st[top]),vis[st[top]]=0,top--;
        top--;p[tot].pb(x);vis[x]=0;
    }
}
signed main(){
    //freopen("1.in","r",stdin);
    //freopen(".out","w",stdout);
    n=read(),m=read();
    for(int i=1;i<=m;i++){x=read(),y=read();e[x].pb(mp(y,i)),e[y].pb(mp(x,i));}
    for(int i=1;i<=n;i++)if(!dfn[i]){tarjan(i,0);}
    cout<<tot<<endl;
    for(int i=1;i<=tot;i++){
        printf("%ld ",p[i].size());
        for(auto j:p[i])printf("%d ",j);
        puts("");
```

```
    }
    return 0;
}
```

3.1.5 点双

```
int n,m,x,y,dfn[maxn],low[maxn],cnt=0,idx=0,st[maxn],top=0,tot=0,vis[maxn];
vector<int>e[maxn],p[maxn];
void tarjan(int x,int fa){
    dfn[x]=low[x]=++idx;
    st[++top]=x;vis[x]=1;
    if(e[x].size()==0){p[++tot].pb(x);}
    for(auto i:e[x])if(i!=fa){
        if(!dfn[i]){
            tarjan(i,x);
            if(dfn[x]<=low[i]){
                ++tot;p[tot].pb(x);
                while(st[top]!=i)p[tot].pb(st[top]),vis[st[top]]=0,top--;
                top--;vis[i]=0;p[tot].pb(i);
            }low[x]=min(low[x],low[i]);
        }
        else low[x]=min(low[x],dfn[i]);
    }
}
signed main(){
    //freopen("1.in","r",stdin);
    //freopen(".out","w",stdout);
    n=read(),m=read();
    for(int i=1;i<=m;i++){x=read(),y=read();if(x==y)
{continue;}e[x].pb(y),e[y].pb(x);}
    for(int i=1;i<=n;i++)if(!dfn[i]){tarjan(i,0);}
    cout<<tot<<endl;
    for(int i=1;i<=tot;i++){
        printf("%ld ",p[i].size());
        for(auto j:p[i])printf("%d ",j);
        puts("");
    }
    return 0;
}
```

3.2 网络流

```
int n,m,x,y,s,t,z;
struct Dinic
{
    int
head[maxn],to[maxn*4],nex[maxn*4],w[maxn*4],cnt=1,dis[maxn],cur[maxn],vis[maxn],s
um,maxflow;
    void add(int x,int y,int z)
{to[++cnt]=y,w[cnt]=z,nex[cnt]=head[x],head[x]=cnt;
    to[++cnt]=x,w[cnt]=0,nex[cnt]=head[y],head[y]=cnt;}
    void clear(){memset(head,0,sizeof head),memset(cur,0,sizeof
cur),cnt=1,maxflow=0;memset(vis,0,sizeof vis);}
    bool bfs()
```

```
    {
        memset(dis,-1,sizeof dis);queue<int>q;
        q.push(s);dis[s]=0;
        for(int i=1;i<=n;i++)cur[i]=head[i];
        while(!q.empty()){
            int x=q.front();q.pop();
            for(int i=head[x];i;i=nex[i]){
                if(dis[to[i]]==-1&&w[i]){q.push(to[i]);dis[to[i]]=dis[x]+1;}
            }
        }
        return dis[t]!=-1;
    }
    int dfs(int x,int flow)
    {
        if(x==t)return flow;int sum=0;
        for(int i=cur[x];i&&flow;i=nex[i]){cur[x]=i;
            if(dis[to[i]]==dis[x]+1&&w[i]){
                int num=dfs(to[i],min(flow,w[i]));
                flow-=num,w[i]-=num;w[i^1]+=num;sum+=num;
            }
        }
        return sum;
    }
    void work()
    {
        while(bfs())maxflow+=dfs(s,inf);
    }
}G;
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    n=read(),m=read();s=read(),t=read();G.cnt=1;
    for(int i=1;i<=m;i++)x=read(),y=read(),z=read(),G.add(x,y,z);
    G.work();cout<<G.maxflow<<endl;
    return 0;
}
```

3.2.1费用流

```
int n,m,s,t,x,y,z,w;
struct SSP
{
    int head[maxn],to[maxn*4],nex[maxn*4],w[maxn*4],c[maxn*4],
    cnt=1,dis[maxn],cur[maxn],vis[maxn],mincost,maxflow;
    void add(int x,int y,int z,int cost)
{to[++cnt]=y,w[cnt]=z,c[cnt]=cost,nex[cnt]=head[x],head[x]=cnt;
    to[++cnt]=x,w[cnt]=0,c[cnt]=-cost,nex[cnt]=head[y],head[y]=cnt;}
    void clear(){memset(head,0,sizeof head),memset(cur,0,sizeof
cur),cnt=1,maxflow=0,mincost=0;memset(vis,0,sizeof vis);}
    bool spfa()
    {
        memset(dis,0x3f,sizeof dis);memcpy(cur,head,sizeof head);
```

```
        queue<int>q;q.push(s);dis[s]=0;vis[s]=1     ;
        while(!q.empty())
        {
            int x=q.front();q.pop();vis[x]=0;
            for(int i=head[x];i;i=nex[i])
                if(dis[to[i]]>dis[x]+c[i]&&w[i])
                {
                    dis[to[i]]=dis[x]+c[i];
                    if(!vis[to[i]])q.push(to[i]),vis[to[i]]=1;
                }
        }
        return dis[t]!=inf;
    }
    int dfs(int x,int flow)
    {
        if(x==t)return flow;
        int sum=0;vis[x]=1;
        for(int i=cur[x];i&&flow;i=nex[i])
        {
            cur[x]=i;
            if(!vis[to[i]]&&w[i]&&dis[to[i]]==dis[x]+c[i])
            {
                int k=dfs(to[i],min(flow,w[i]));
                flow-=k,sum+=k;w[i]-=k,w[i^1]+=k;mincost+=k*c[i];
            }
        }vis[x]=0;
        return sum;
    }
    void work()
    {
        while(spfa())maxflow+=dfs(s,inf);
    }
}G;
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    n=read(),m=read(),s=read(),t=read();G.cnt=1;
    for(int i=1;i<=m;i++)x=read(),y=read(),z=read(),w=read(),G.add(x,y,z,w);
    G.work();cout<<G.maxflow<<" "<<G.mincost<<endl;
    return 0;
}
```

3.2.2 有源汇上下界网络流

```
const int maxn=1e6+10;
int n,m,s,t,tot,cnt=0,x,y,z,head[maxn],p;
struct node
{
    int v,nex;long long w;
}e[maxn];
const int inf=1<<29;
void add(int u,int v,int w)
{e[++cnt].v=v,e[cnt].w=w,e[cnt].nex=head[u];head[u]=cnt;}
int d[maxn],cur[maxn],in[maxn],out[maxn],g[maxn],c[maxn];
```

```
bool bfs(int s,int t)
{
    memset(d,-1,sizeof d);
    queue<int>q;q.push(s);d[s]=0;cur[s]=head[s];
    while(!q.empty())
    {
        int u=q.front();q.pop();
        for(int i=head[u];i;i=e[i].nex)
            if(e[i].w&&d[e[i].v]==-1)
            {
                int v=e[i].v;
                d[v]=d[u]+1;q.push(v);cur[v]=head[v];
                if(v==t)
                    return 1;
            }
    }
    return 0;
}
long long dfs(int u,long long limit,int t)
{
    if(u==t)return limit;
    int flow=0;
    for(int i=cur[u];i&&flow<limit;i=e[i].nex)
    {
        cur[u]=i;int v=e[i].v;
        if(e[i].w&&d[v]==d[u]+1)
        {
            int f=dfs(v,min(e[i].w,limit-flow),t);
            if(!f)d[v]=-1;e[i].w-=f,e[i^1].w+=f;flow+=f;
        }
    }
    return flow;
}
long long dinic(int s,int t)
{
    long long maxflow=0,flow=0;cur[s]=head[s];
    while(bfs(s,t))
    {
        while(flow=dfs(s,inf,t))
            maxflow+=flow;
    }
    return maxflow;
}
int main()
{
    while(~scanf("%d %d",&n,&m))
    {
        memset(e,0,sizeof e);int
s=n+m+1,t=s+1,s2=t+1,t2=s2+1;cnt=1;memset(cur,0,sizeof cur);
        memset(head,0,sizeof head);memset(in,0,sizeof in);memset(out,0,sizeof
out);int sum=0;
        for(int
i=1;i<=m;i++)g[i]=read(),add(i+n,t,inf),add(t,i+n,0),in[t]+=g[i],out[i+n]+=g[i];
        for(int i=1;i<=n;i++)
        {
            c[i]=read(),p=read();add(s,i,p),add(i,s,0);
```

```
            for(int j=1;j<=c[i];j++)
            {
                int tt=read(),l=read(),r=read();tt++;add(i,tt+n,r-
l);add(tt+n,i,0);out[i]+=l,in[tt+n]+=l;
            }
        }//cout<<sum<<endl;
        int ll=0;
        for(int i=1;i<=t;i++)
            if(in[i]>out[i])add(s2,i,in[i]-out[i]),ll+=in[i]-out[i],add(i,s2,0);
            else add(i,t2,out[i]-in[i]),add(t2,i,0);
        add(t,s,inf);add(s,t,0);
        long long flow1=dinic(s2,t2);//cout<<ll<<" "<<flow1<<' '<<cnt<<endl;
        if(flow1!=ll){cout<<"-1\n\n";continue;}long long ans=e[cnt].w;
        e[cnt].w=0;e[cnt-1].w=0;
        long long flow2=dinic(s,t);
        cout<<ans+flow2<<endl<<endl;
    }

}
```

3.3 2-sat

> ### 怎么求解 2-SAT 问题?
>
> **使用强连通分量。** 对于每个变量 x，我们建立两个点：$x, \neg x$ 分别表示变量 x 取 `true` 和取 `false`。所以，**图的节点个数是两倍的变量个数。在存储方式上，可以给第 i 个变量标号为 i，其对应的反值标号为 $i+n$。**对于每个同学的要求 $(a \vee b)$，转换为 $\neg a \to b \wedge \neg b \to a$。对于这个式子，可以理解为：「若 a 假则 b 必真，若 b 假则 a 必真」然后按照箭头的方向建有向边就好了。综上，我们这样对上面的方程建图：
>
> | 原式 | 建图 |
> | --- | --- |
> | $\neg a \vee b$ | $a \to b \wedge \neg b \to \neg a$ |
> | $a \vee b$ | $\neg a \to b \wedge \neg b \to a$ |
> | $\neg a \vee \neg b$ | $a \to \neg b \wedge b \to \neg a$ |
>
> 于是我们得到了这么一张图：

```
const int maxn=2000005;
struct edge{int v,nex;}e[maxn];
int
n,m,head[maxn],x,y,f1,f2,dfn[maxn],id[maxn],low[maxn],idx=0,top=0,st[maxn],vis[ma
xn],cnt=0,tot;
void add(int u,int v){e[++cnt].v=v,e[cnt].nex=head[u],head[u]=cnt;}
void tarjan(int x)
{
    dfn[x]=low[x]=++idx;st[++top]=x;vis[x]=1;
    for(int i=head[x];i;i=e[i].nex)
    {
        int v=e[i].v;
        if(!dfn[v])tarjan(v),low[x]=min(low[v],low[x]);
        else if(vis[v])low[x]=min(low[x],dfn[v]);
    }
```

```
    if(low[x]==dfn[x])
    {
        tot++;
        while(st[top+1]!=x)
            id[st[top]]=tot,vis[st[top]]=0,top--;
    }
}
int main()
{
    n=read();m=read();
    for(int i=1;i<=m;i++)
    {
        x=read(),f1=read(),y=read(),f2=read();
        if(f1&&f2)add(x,y+n),add(y,x+n);if(f1&&!f2)add(x,y),add(y+n,x+n);
        if(!f1&&f2)add(x+n,y+n),add(y,x);if(!f1&&!f2)add(x+n,y),add(y+n,x);
    }
    for(int i=1;i<=n+n;i++)if(!id[i])tarjan(i);
    for(int i=1;i<=n;i++)
        if(id[i]==id[i+n])puts("IMPOSSIBLE"),exit(0);
    puts("POSSIBLE");
    for(int i=1;i<=n;i++)
        if(id[i]>id[i+n])printf("1 ");else printf("0 ");
    return 0;
}
```

3.4 KM

给定一张二分图，左右部均有 n 个点，共有 m条带权边，且保证有完美匹配。

求一种完美匹配的方案，使得最终匹配边的边权之和最大。

```
int n,m,d[maxn][maxn],mat[maxn],Min[maxn],p,dx[maxn],dy[maxn],pre[maxn],x,y;
bool vis[maxn];
void match(int x)
{
    int y=0,num,id;
    memset(pre,0,sizeof pre);mat[y]=x;
    for(int i=1;i<=n;i++)Min[i]=inf;
    while(1)
    {
        x=mat[y];vis[y]=1;num=inf;
        for(int i=1;i<=n;i++)
        {
            if(vis[i])continue;
            if(Min[i]>dx[x]+dy[i]-d[x][i])
                Min[i]=dx[x]+dy[i]-d[x][i],pre[i]=y;
            if(Min[i]<num)num=Min[i],id=i;
        }
        for(int i=0;i<=n;i++)
        {
            if(vis[i])dx[mat[i]]-=num,dy[i]+=num;
            else Min[i]-=num;
        }y=id;if(mat[y]==-1)break;
    }
    while(y)mat[y]=mat[pre[y]],y=pre[y];
```

```
}
int KM()
{
    memset(mat,-1 ,sizeof mat);
    memset(dx,0,sizeof dx);memset(dy,0,sizeof dy);
    for(int i=1;i<=n;i++)
        memset(vis,0,sizeof vis),match(i);
    int res=0;
    for(int i=1;i<=n;i++)
        if(mat[i]!=-1)res+=d[mat[i]][i];
    return res;
}
signed main()
{
    n=read();m=read();memset(d,0x80,sizeof d);
    for(int i=1;i<=m;i++)x=read(),y=read(),d[x][y]=read();
    printf("%lld\n",KM());
    for(int i=1;i<=n;i++)printf("%lld ",mat[i]);
    return 0;
}
```

3.5 匈牙利

```
int n,m,e,ans;
int vis[N][N];
int ask[N],matched[N];

inline bool found(int x){ //dfs找增广路
    for (int i = 1 ; i <= m ; i++)
      if (vis[x][i]){
        if (ask[i])
            continue;
        ask[i] = 1;
        if (!matched[i] || found(matched[i])) {
            matched[i] = x ;
            return true;
        }
      }
    return false;
}

inline void match(){
    int cnt = 0;//cnt是计数器
    memset(matched,0,sizeof(matched));
    for (int i = 1 ; i <= n ; i++){
      memset(ask,0,sizeof(ask));
      if (found(i))
        cnt++;  //找到了就加1
    }
    ans = cnt;
}
//从这里向下看起
int main(){
    scanf("%d%d%d",&n,&m,&e);//结点个数分别为n,m，边数为e
    for (int i = 1 ; i <= e ; i++){
```

```
      int x,y;
      scanf("%d%d",&x,&y);
      vis[x][y] = 1;
    }
    match();///匈牙利算法，见上
    printf("%d \n",ans);
    return 0;
}
```

3.6 差分约束

差分约束问题可以转化为最短路或最长路问题，所以两种转化也就形成了两种不同的连边方法。

1. 连边后求最短路
   变形为 $x_j \leq x_i + k$ 即从 i 到 j 连一条边权为 k 的边。加入超级源点后求最短路，得到 $xi \leq 0$ 所有 x 最大解。
2. 连边后求最长路
   变形为 $x_i \geq x_j - k$，即从j 到 i 连一条边权为 $-k$ 的边。加入超级源点后求最长路，得到$xi \geq 0$ 所有 x 最小解。

3.7 虚树

```
int n,m,d[maxn],f[maxn]
[21],x,y,z,dfn[maxn],idx=0,a[maxn],st[maxn],top,minv[maxn],head[maxn],cnt;
bool flag[maxn];
struct edge{int v,w;edge(int x,int y):v(x),w(y){}};
struct node{int v,nex;}e2[maxn];
vector<edge>e[maxn];
void add(int x,int y){e2[++cnt].v=y,e2[cnt].nex=head[x];head[x]=cnt;}
void dfs(int x,int fa)
{
    d[x]=d[fa]+1;f[x][0]=fa;dfn[x]=++idx;
    for(int i=1;i<=20;i++)f[x][i]=f[f[x][i-1]][i-1];
    for(auto i:e[x])
    {
        int v=i.v;if(v==fa)continue;
        minv[v]=min(minv[x],i.w);dfs(v,x);
    }
}
int Lca(int x,int y)
{
    if(d[x]<d[y])swap(x,y);
    for(int i=20;i>=0;i--)if(d[f[x][i]]>=d[y])x=f[x][i];
    if(x==y)return x;
    for(int i=20;i>=0;i--)if(f[x][i]!=f[y][i])x=f[x][i],y=f[y][i];
    return f[x][0];
}
bool cmp(int x,int y){return dfn[x]<dfn[y];}
void ins(int x)
{
    if(top==0){st[top=1]=x;return;}
    int lca=Lca(st[top],x);
```

```
    while(top>1&&d[lca]<d[st[top-1]])
        add(st[top-1],st[top]),--top;
    if(d[lca]<d[st[top]])add(lca,st[top--]);
    if((!top)||(st[top]!=lca))st[++top]=lca;st[++top]=x;
}
int dfs1(int x)
{
    int sum=0,num;
    for(int i=head[x];i;i=e2[i].nex)sum+=dfs1(e2[i].v);
    if(flag[x])num=minv[x];
    else num=min(minv[x],sum);
    head[x]=0;flag[x]=0;return num;
}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    memset(minv,0x3f,sizeof minv);
    n=read();
    for(int i=1;i<n;i++)

x=read(),y=read(),z=read(),e[x].push_back(edge(y,z)),e[y].push_back(edge(x,z));
    dfs(1,0);//cout<<Lca(1,2)<<endl;
    m=read();
    while(m--)
    {
        x=read();top=0;
        for(int i=1;i<=x;i++)a[i]=read(),flag[a[i]]=1;
        sort(a+1,a+x+1,cmp);
        for(int i=1;i<=x;i++)ins(a[i]);//,cout<<st[1]<<" "<<top<<endl;
        while(--top)add(st[top],st[top+1]);//cout<<st[1]<<endl;
        printf("%lld\n",dfs1(st[1]));cnt=0;
    }
    return 0;
}
```

3.8 欧拉路径

```
int
n,m,x,y,head[maxn],in[maxn],p[maxn],s=1,t,c=0,st[maxn],tot=0;vector<int>e[maxn];
void dfs(int x){

    for(int i=p[x];i<e[x].size();i=p[x]){
        int v=e[x][i];p[x]=i+1;dfs(v);
    }
    st[++tot]=x;
}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    n=read(),m=read();
    for(int i=1;i<=m;i++)x=read(),y=read(),in[y]++,in[x]--,e[x].pb(y);
    for(int i=1;i<=n;i++)sort(e[i].begin(),e[i].end()),p[i]=0;
    for(int i=1;i<=n;i++)if(in[i]>0)c++,t=i;
```

```
    else if(in[i]<0)c++,s=i;
    if(c>2)puts("No"),exit(0);
    dfs(s);
    for(int i=tot;i>=1;i--)printf("%d ",st[i]);
    return 0;
}
```

3.9 最小割树

最小割树=分治+最小割（最大流）

给定一个 n个点 m 条边的无向连通图，多次询问两点之间的最小割。

```
void init(){
    for(int i=2;i<=cnt;i+=2)w[i]=w[i^1]=(w[i]+w[i^1])/2;
}
void add(int x,int y,int z){to[++cnt]=y,nex[cnt]=head[x],w[cnt]=z;head[x]=cnt;}
bool bfs(){
    for(int i=1;i<=n;i++)dis[i]=-1,cur[i]=head[i];
    queue<int>q;q.push(s);dis[s]=0;
    while(!q.empty()){
        int x=q.front();q.pop();
        for(int i=head[x];i;i=nex[i])
            if(dis[to[i]]==-1&&w[i])dis[to[i]]=dis[x]+1,q.push(to[i]);
    }if(dis[t]==-1)return 0;return 1;
}
int dfs(int x,int flow){
    if(x==t)return flow;
    int sum=0;
    for(int i=cur[x];i&&flow;i=nex[i]){
        cur[x]=i;
        if(dis[to[i]]==dis[x]+1&&w[i]){
            int k=dfs(to[i],min(w[i],flow));
            w[i]-=k,w[i^1]+=k,flow-=k;sum+=k;
        }
    }return sum;
}
int dinic(){
    int sum=0;
    while(bfs())sum+=dfs(s,inf);
    return sum;
}
void build(int l,int r){
    if(l==r)return;//cout<<l<<" "<<r<<endl;
    init();
    s=a[l],t=a[r];//cout<<s<<" "<<t<<endl;
    int x=dinic();//cout<<x<<endl;
    e[s].pb(mp(t,x));e[t].pb(mp(s,x));
    tmp1.clear(),tmp2.clear();
    for(int i=l;i<=r;i++)
        if(dis[a[i]]==-1)tmp1.pb(a[i]);
        else tmp2.pb(a[i]);
    int u=tmp1.size(),v=tmp2.size(),g=l-1;
    for(int i=0;i<u;i++)a[++g]=tmp1[i];
    int p=g;
```

```
    for(int i=0;i<v;i++)a[++g]=tmp2[i];
    build(l,p),build(p+1,r);
}
void dfs(int x,int fa,int z){
    for(auto i:e[x])if(i.fi^fa)
        ans[i.fi][z]=min(ans[x][z],i.se),dfs(i.fi,x,z);
}
signed main(){
    //freopen("1.in","r",stdin);
    //freopen(".out","w",stdout);
    n=read(),m=read();
    for(int i=1;i<=m;i++)x=read(),y=read(),z=read(),add(x,y,z),add(y,x,z);
    for(int i=1;i<=n;i++)a[i]=i;
    build(1,n);
    for(int i=1;i<=n;i++)ans[i][i]=1e9,dfs(i,0,i);
    q=read();
    for(int i=1;i<=q;i++){
        x=read(),y=read();printf("%d\n",ans[x][y]);
    }
    return 0;
}
```

4 计算几何与数学

4.1 凸包

```
int n,tot=0;double x,y,res=0;
struct node{
    double x,y;
}a[maxn],st[maxn];
bool cmp(node a,node b){if(a.x==b.x)return a.y<b.y;return a.x<b.x;}
double cross(node a,node b){return a.x*b.y-b.x*a.y;}
bool check(node a,node b,node c,node d){b.x-=a.x,b.y-=a.y;d.x-=c.x,d.y-
=c.y;if(cross(b,d)>=0)return 1;return 0;}
double getdis(node a,node b){return sqrt((a.x-b.x)*(a.x-b.x)+(a.y-b.y)*(a.y-
b.y));}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    n=read();
    for(int i=1;i<=n;i++)scanf("%lf%lf",&a[i].x,&a[i].y);
    sort(a+1,a+n+1,cmp);st[++tot]=a[1];
    for(int i=2;i<=n;i++){
        while(tot>1&&check(st[tot-1],st[tot],st[tot-1],a[i]))tot--;
        st[++tot]=a[i];
    }st[++tot]=a[n];int p=tot;
    for(int i=n-1;i>=1;i--){
        while(tot>p&&check(st[tot-1],st[tot],st[tot-1],a[i]))tot--;
        st[++tot]=a[i];
    }st[++tot]=a[1];
    for(int i=1;i<tot;i++)res+=getdis(st[i],st[i+1]);
    printf("%.2lf",res);
```

```
    return 0;
}
```

4.1.1旋转卡壳（求凸包直径）

```
int n,m,x,y,top=0,ans;
pi a[maxn],st[maxn];
int cross(pi a,pi b){return a.fi*b.se-a.se*b.fi;}
int get(pi a,pi b,pi c,pi d){
    a=mp(a.fi-b.fi,a.se-b.se),c=mp(c.fi-d.fi,c.se-d.se);
    return cross(a,c);
}
int dis(pi a,pi b){return (a.fi-b.fi)*(a.fi-b.fi)+(a.se-b.se)*(a.se-b.se);}
signed main(){
    n=read();
    for(int i=1;i<=n;i++){
        x=read(),y=read(),a[i]=mp(x,y);
    }sort(a+1,a+n+1);
    for(int i=1;i<=n;i++){
        while(top>1&&get(st[top-1],st[top],st[top-1],a[i])<=0)top--;
        st[++top]=a[i];
    }int p=top;
    for(int i=n;i>=1;i--){
        while(top>p&&get(st[top-1],st[top],st[top-1],a[i])<=0)top--;
        st[++top]=a[i];
    }
    top--;
    int j=1;if(top==2){cout<<dis(a[1],a[n]);exit(0);}
    for(int i=1;i<=top;i++){
        while(get(st[i],st[i+1],st[i],st[j])<=get(st[i],st[i+1],st[i],st[j+1]))
            j=j%top+1;
        ans=max(ans,dis(st[i],st[j]));
        ans=max(ans,dis(st[i+1],st[j]));
    }cout<<ans<<endl;
    return 0;
}
```

斜率优化dp

```
//S[n]=ΣC[i]+1, dp[i]=min(dp[j]+(S[i]-(S[j]+L+1))^2), ++L
//dp[i]=S[i]^2-2*S[i]*L+dp[j]+(S[j]+L)^2-2S[i]*S[j]
//(2*S[i]) * S[j] + (dp[i]-S[i]^2+2S[i]L)=(dp[j]+(S[j]+L)^2)
//   k     *  x   +           b          =        y
int n,l,h,top=0,s[maxn],f[maxn],st[maxn];
inline int X(int x){return s[x];}
inline int Y(int x){return f[x]+(s[x]+l)*(s[x]+l);}
inline long double getk(int i,int j){return (long double)(Y(j)-Y(i))/(X(j)-
X(i));}
signed main()
{
    n=read(),l=read();l++;st[++top]=0;h=1;
    for(int i=1;i<=n;i++)s[i]=read(),s[i]+=s[i-1]+1;
    for(int i=1;i<=n;i++)
    {
```

```
        while(h<top&&getk(st[h],st[h+1])<=2*s[i])h++;
        int j=st[h];f[i]=f[j]+(s[i]-s[j]-l)*(s[i]-s[j]-l);
        while(h<top&&getk(st[top-1],st[top])>=getk(st[top],i))top--;
        st[++top]=i;
    }
    cout<<f[n];
    return 0;
}
```

4.2 最小圆覆盖

```
const double eps=1e-12;
struct node{double x,y;}a[maxn],o;
double c,x,y,r;
int n;
double dis(node p1,node p2)
{return sqrt((p1.x-p2.x)*(p1.x-p2.x)+(p1.y-p2.y)*(p1.y-p2.y));}
void getcircle(node p1,node p2,node p3)
{
    double a=p1.x-p2.x,b=p1.y-p2.y,c=p1.x-p3.x,d=p1.y-p3.y;
    double e=((p1.x*p1.x-p2.x*p2.x)-(p2.y*p2.y-p1.y*p1.y))/2;
    double f=((p1.x*p1.x-p3.x*p3.x)-(p3.y*p3.y-p1.y*p1.y))/2;
    o.x=(b*f-d*e)/(b*c-a*d);o.y=(c*e-a*f)/(b*c-a*d);r=dis(o,p1);
}
signed main()
{
    n=read();
    for(int i=1;i<=n;i++)scanf("%lf%lf",&a[i].x,&a[i].y);
    random_shuffle(a+1,a+n+1);
    o=a[1];r=0;
    for(int i=2;i<=n;i++){
        if(dis(o,a[i])>r+eps){
            o=a[i];r=0;
            for(int j=1;j<i;j++){
                if(dis(o,a[j])>r+eps){
                    o.x=(a[i].x+a[j].x)/2;
                    o.y=(a[i].y+a[j].y)/2;r=dis(o,a[j]);
                    for(int k=1;k<j;k++)
                        if(dis(o,a[k])>r+eps)
                            getcircle(a[i],a[j],a[k]);
                }
            }
        }
    }
    printf("%.10lf\n%.10lf %.10lf",r,o.x,o.y);
    return 0;
}
```

4.3 多项式NTT全家桶

NTT板子

```
const int mod=998244353,G=3,maxn=4e6+5;
int n,m,len=1,a[maxn],b[maxn],x,y,r[maxn],s=0,Ginv;
```

```
int Pow(int x,int y){int res=1;while(y)
{if(y&1)res=res*x%mod;x=x*x%mod;y=y/2;}return res;}
void NTT(int *A,int op){
    for(int i=0;i<len;i++)if(i<r[i])swap(A[i],A[r[i]]);
    for(int i=1;i<len;i=i*2){
        int o=i*2;int w=Pow((op==1)?G:Ginv,(mod-1)/o);
        for(int j=0;j<len;j+=o){
            int res=1;
            for(int k=0;k<i;k++){
                int x=A[j+k],y=res*A[i+j+k]%mod;
                A[j+k]=(x+y)%mod,A[i+j+k]=(x-y+mod)%mod;res=res*w%mod;
            }
        }
    }
}
signed main(){
    n=read();m=read();len=1;Ginv=Pow(G,mod-2);
    for(int i=0;i<=n;i++)a[i]=read();
    for(int i=0;i<=m;i++)b[i]=read();
    while(len<=n+m)len=len*2,s++;
    for(int i=0;i<len;i++)
        for(int j=0;j<s;j++)if((1<<j)&i)r[i]+=(1<<(s-j-1));
    NTT(a,1),NTT(b,1);
    for(int i=0;i<len;i++)a[i]=a[i]*b[i]%mod;
    NTT(a,-1);int inv=Pow(len,mod-2);
    for(int i=0;i<=n+m;i++)printf("%lld ",a[i]*inv%mod);
    return 0;
}
```

多项式求ln

```
const int maxn=4e5+5,mod=998244353,G=3;//G是原根
int n,m,len=1,r[maxn],b[3][maxn],a[maxn],sum=0,a1[maxn],a2[maxn],a3[maxn];
inline int Pow(int a,int b)
{
    int res=1;
    while(b)
    {
        if(b&1)res=res*a%mod;a=a*a%mod;b>>=1;
    }return res%mod;
}
inline void NTT(int *A,int len,int opt){
    for(int i=0;i<len;i++)
        if(i<r[i])swap(A[i],A[r[i]]);
    for(int i=1;i<len;i<<=1){
        int W=Pow(G,(mod-1)/(i<<1));
        if(opt==-1)W=Pow(Pow(G,mod-2),(mod-1)/(i<<1));
        for(int j=0;j<len;j+=i*2){
            int w=1;
            for(int k=0;k<i;k++,w=(w*W)%mod){
                int x=A[j+k],y=w*A[i+j+k]%mod;
                A[j+k]=(x+y)%mod;A[i+j+k]=((x-y)%mod+mod)%mod;
            }
        }
    }
```

```
    if(opt==-1)
    {
        int inv=Pow(len,mod-2);
        for(int i=0;i<len;i++)A[i]=A[i]*inv%mod;
    }
}
int X[maxn],Y[maxn],L,tag=0;
inline void mul(int *x,int *y,int len)
{
    memset(X,0,sizeof X);memset(Y,0,sizeof Y);
    for(int i=0;i<(len/2);i++)X[i]=x[i],Y[i]=y[i];
    NTT(X,len,1),NTT(Y,len,1);
    for(int i=0;i<len;i++)X[i]=X[i]*Y[i]%mod;
    NTT(X,len,-1);for(int i=0;i<len;i++)x[i]=X[i];
}
inline void getinv(int *a,int *B,int len)
{
    len=2,L=1,tag=0,sum=1;
    tag=0;b[tag][0]=Pow(a[0],mod-2);tag=1;
    while(L<=n*2)
    {
        if(L<=n*2)
            for(int i=0;i<len;i++)r[i]=(r[i>>1]>>1)|((i&1)<<(sum-1));
        for(int i=0;i<len;i++)b[tag][i]=b[tag^1][i]*2%mod;
        mul(b[tag^1],b[tag^1],len),mul(b[tag^1],a,len);
        for(int i=0;i<len;i++)b[tag][i]=(b[tag][i]-b[tag^1][i]+mod)%mod;
        L=L*2;len=len*2,sum++;tag^=1;
    }
    for(int i=0;i<m;i++)
        B[i]=b[tag^1][i]%mod;
}
inline void getdao(int *a,int *B,int len)
{
    for(int i=1;i<len;i++)B[i-1]=a[i]*i%mod;B[len-1]=0;
}
inline void getjifen(int *a,int *B,int len)
{
    for(int i=1;i<len;i++)B[i]=a[i-1]*Pow(i,mod-2)%mod;B[0]=0;
}
signed main()
{
    n=read();n--;
    for(int i=0;i<=n;i++)a[i]=(read()+mod)%mod;
    for(m=1;m<=n;m<<=1);getdao(a,a1,m);getinv(a,a2,2);
    sum=0;len=1;while(len<=m)len=len*2,sum++;
    for(int i=0;i<len;i++)r[i]=(r[i>>1]>>1)|((i&1)<<(sum-1));
    NTT(a1,len,1),NTT(a2,len,1);
    for(int i=0;i<=len;i++)a2[i]=a1[i]*a2[i]%mod;
    NTT(a2,len,-1);getjifen(a2,a3,m);
    for(int i=0;i<=n;i++)printf("%d ",a3[i]);

    return 0;
}
```

分治NTT

给定序列 $g_{1\dots n-1}$，求序列 $f_{0\dots n-1}$。

其中 $f_i=\sum_{j=1}^{i} f_{i-j}g_j$，边界为 $f_0=1$。

答案对 998244353 取模。

```
const int maxn=4e5+5,M=34005,mod=998244353,G=3;
int n,m=1,G1,s=0,a[maxn],b[maxn],c[maxn],d[maxn],r[maxn];
int Pow(int x,int y){int res=1;while(y)
{if(y&1)res=res*x%mod;x=x*x%mod;y=y/2;}return res;}
struct poly{
    vector<int>a;
    int &operator[](int x){return a[x];};
    int size(){return a.size();}
    void resize(int x){a.resize(x);}
    void print(){for(auto y:a)cout<<y<<" ";puts("");}
}f,g;
void NTT(poly &A,int len,int op){
    for(int i=0;i<len;i++)if(i<r[i])swap(A[i],A[r[i]]);
    for(int i=1;i<len;i=i*2){
        int o=i*2;int w=Pow((op==0)?G:G1,(mod-1)/o);
        for(int j=0;j<len;j+=o){
            int res=1;
            for(int k=0;k<i;k++){
                int x=A[j+k],y=res*A[i+j+k]%mod;
                A[j+k]=(x+y)%mod,A[i+j+k]=(x-y+mod)%mod;res=res*w%mod;
            }
        }
    }
    if(op==1){
        int inv=Pow(len,mod-2);
        for(int i=0;i<len;i++)A[i]=A[i]*inv%mod;
    }
}
int get(int n){int s=0;while((1<<s)<n)s++;return s;}
poly operator*(poly a,poly b){
    int n=a.size()+b.size()-1;poly c;
    int z=get(n);int m=1<<z;c.resize(m);
    for(int i=1;i<m;i++)r[i]=(r[i>>1]>>1)|((i&1)<<(z-1));
    a.resize(m),b.resize(m);NTT(a,m,0),NTT(b,m,0);
    for(int i=0;i<m;i++)c[i]=a[i]*b[i]%mod;
    NTT(c,m,1);c.resize(n);return c;
}
void solve(int l,int r){
    if(l==r)return;
    solve(l,mid);
    poly a,b;a.resize(mid-l+1);b.resize(r-l+1);
    for(int i=l;i<=mid;i++)a[i-l]=f[i];
    for(int i=0;i<r-l+1;i++)b[i]=g[i];a=a*b;
    for(int i=mid+1;i<=r;i++)f[i]=(f[i]+a[i-l])%mod;
    solve(mid+1,r);
}
signed main(){
//  freopen("1.in","r",stdin);
```

```
    //freopen(".out","w",stdout);
    G1=Pow(3,mod-2);
    n=read();f.resize(n),g.resize(n);
    for(int i=1;i<n;i++)g[i]=read();f[0]=1;
    solve(0,n-1);
    for(int i=0;i<n;i++)printf("%lld ",f[i]);
    return 0;
}
```

4.4莫比乌斯反演

```
int n,m,mo[maxn],p[maxn],tot=0,vis[maxn],a,b;
long long sum[maxn],ans,d[maxn];
void get_mo(int x)
{
    mo[1]=1;
    for(int i=2;i<=x;i++)
    {
        if(!vis[i])
        {
            mo[i]=-1;p[++tot]=i;
        }
        for(int j=1;j<=tot&&i*p[j]<=x;j++)
        {
            vis[i*p[j]]=1;
            if(i%p[j]==0)break;
            else mo[i*p[j]]=-mo[i];
        }
    }

    for(int i=1;i<=tot;i++)
    {
        int g=p[i];
        for(int j=g;j<=x;j=j+g)
        {
            d[j]+=mo[j/g];
        }
    }for(int i=1;i<=x;i++)sum[i]=sum[i-1]+d[i];
}
signed main()
{
    n=read();
    get_mo(10000001);
    for(int i=1;i<=n;i++)
    {
        a=read(),b=read(),ans=0;
        for(int l=1,r;l<=min(a,b);l=r+1)
        {
            r=min(a/(a/l),b/(b/l));
            ans+=(sum[r]-sum[l-1])*(a/(r))*(b/(r));
        }
        printf("%lld\n",ans);
    }
    return 0;
```

```
}
```

4.5 杜教筛

```
int T,mo[maxn],n,vis[maxn],p[maxn],sm[maxn],ph[maxn];
long long sp[maxn];
map<int,int>ansmo;
map<int,long long>ansphi;
void get(int x)
{
    int tot=0;mo[1]=1;ph[1]=1;sp[1]=1,sm[1]=1;
    for(int i=2;i<=x;i++)
    {
        if(!vis[i])mo[i]=-1,ph[i]=i-1,vis[i]=i,p[++tot]=i;
        for(int j=1;j<=tot;j++)
        {
            if(i*p[j]>x)break;
            if(i%p[j]==0)mo[i*p[j]]=0;
            else mo[i*p[j]]=-mo[i];
            if(p[j]<vis[i])vis[i*p[j]]=p[j],ph[i*p[j]]=ph[i]*ph[p[j]];
            else if(p[j]==vis[i])vis[i*p[j]]=p[j],ph[i*p[j]]=ph[i]*p[j];
            //cout<<j<<endl;
        }
        sp[i]=sp[i-1]+ph[i],sm[i]=sm[i-1]+mo[i];
    }
}
inline int getmo(int x)
{//cout<<x<<endl;
    if(x<=5000000)return sm[x];
    if(ansmo[x])return ansmo[x];
    int res=1;//cout<<"A"<<x<<endl;
    for(unsigned int l=2,r;l<=x;l=r+1)
    {
        r=x/(x/l);
        res-=(r-l+1)*getmo(x/l);
    }
    return ansmo[x]=res;
}
inline long long getph(long long x)
{
    if(x<=5000000)return sp[x];
    if(ansphi[x])return ansphi[x];
    long long res=1ll*(x+1)*x/2;
    for(unsigned int l=2,r;l<=x;l=r+1)
    {
        r=x/(x/l);
        res-=1ll*(r-l+1)*getph(x/l);
    }
    return ansphi[x]=res;
}
signed main()
{
    T=read();get(5000000);
    while(T--)
        n=read(),cout<<getph(n)<<" "<<getmo(n)<<endl;
```

```
    return 0;
}
```

4.6高斯消元

```
int T,mo[maxn],n,vis[maxn],p[maxn],sm[maxn],ph[maxn];
long long sp[maxn];
map<int,int>ansmo;
map<int,long long>ansphi;
void get(int x)
{
    int tot=0;mo[1]=1;ph[1]=1;sp[1]=1,sm[1]=1;
    for(int i=2;i<=x;i++)
    {
        if(!vis[i])mo[i]=-1,ph[i]=i-1,vis[i]=i,p[++tot]=i;
        for(int j=1;j<=tot;j++)
        {
            if(i*p[j]>x)break;
            if(i%p[j]==0)mo[i*p[j]]=0;
            else mo[i*p[j]]=-mo[i];
            if(p[j]<vis[i])vis[i*p[j]]=p[j],ph[i*p[j]]=ph[i]*ph[p[j]];
            else if(p[j]==vis[i])vis[i*p[j]]=p[j],ph[i*p[j]]=ph[i]*p[j];
            //cout<<j<<endl;
        }
        sp[i]=sp[i-1]+ph[i],sm[i]=sm[i-1]+mo[i];
    }
}
inline int getmo(int x)
{//cout<<x<<endl;
    if(x<=5000000)return sm[x];
    if(ansmo[x])return ansmo[x];
    int res=1;//cout<<"A"<<x<<endl;
    for(unsigned int l=2,r;l<=x;l=r+1)
    {
        r=x/(x/l);
        res-=(r-l+1)*getmo(x/l);
    }
    return ansmo[x]=res;
}
inline long long getph(long long x)
{
    if(x<=5000000)return sp[x];
    if(ansphi[x])return ansphi[x];
    long long res=1ll*(x+1)*x/2;
    for(unsigned int l=2,r;l<=x;l=r+1)
    {
        r=x/(x/l);
        res-=1ll*(r-l+1)*getph(x/l);
    }
    return ansphi[x]=res;
}
signed main()
{
    T=read();get(5000000);
    while(T--)
```

```
        n=read(),cout<<getph(n)<<" "<<getmo(n)<<endl;
    return 0;
}
```

4.7 中国剩余定理

```
int x,y,a,b,n,A,B;
void exgcd(int a,int b){
    if(!b){x=1,y=0;return;}
    exgcd(b,a%b);int t=x;x=y;y=t-a/b*y;
}
void excrt(int &a1,int &b1,int a2,int b2){
     int d=__gcd(a1,a2);
     if((b2-b1)%d!=0){puts("Impossible");return;}
     int z=(b2-b1)/d,p1=a1/d,p2=a2/d,mod=a1/d*a2;
     exgcd(p1,p2);b1=(z*x*a1+b1)%mod;a1=mod;
     b1+=mod;b1%=mod;
}
signed main(){
    //freopen("1.in","r",stdin);
    n=read();A=read(),B=read();
    for(int i=2;i<=n;i++){
        a=read(),b=read();
        excrt(A,B,a,b);
    }cout<<(ll)B<<endl;
    return 0;
}
```

4.8 大步小步

```
int n,m,x,y,T,mod,a,b;map<int,int>f;
int gcd(int x,int y){if(!y)return x;return gcd(y,x%y);}
int bsgs(int a,int b,int mod,int o){
    int s=1,g=o,t=sqrt(mod),x;f.clear();
    for(int i=1;i<=t;i++)s=s*a%mod,x=s*b%mod,f[x]=i;
    for(int i=1;i<=t+2;i++){g=g*s%mod;
        if(f.find(g)!=f.end())return i*t-f[g];}
    return -1e9;
}
int exbsgs(int a,int b,int mod){
    int g=gcd(mod,a);int o=1,cnt=0;b=b%mod;
    if(b==1)return 0;
    while(g>1){
        if(b%g){return -1e9;}cnt++;
        b=b/g;mod=mod/g;o=o*a/g%mod;
        if(o==b){return cnt;}g=gcd(a,mod);
    }
    return cnt+bsgs(a,b,mod,o);
}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    while(1){
```

```
        a=read(),mod=read(),b=read();
        if(!mod)break;int x=exbsgs(a,b,mod);
        if(x<0)puts("No Solution");
        else print(x),puts("");
    }
    return 0;
}
```

4.9 扩欧

```
int Pow(int x,int y)
{
    int res=1;while(y){if(y&1)res=res*x%m;y=y/2;x=x*x%m;}return res;
}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    cin>>n>>m;x=phi=m;
    for(int i=2;i*i<=m;i++)
        if(x%i==0)
        {
            phi=phi-phi/i;
            while(x%i==0)x/=i;
        }
    if(x>1)phi=phi-phi/x;
    b=read();if(F)b+=phi;
    cout<<Pow(n,b)%m;
    return 0;
}
```

4.10 FWT

```
int n,a[maxn],b[maxn];
struct FWT{
    int f[maxn];
    void AND(int x){
        for(int o=2,k=1;o<=n;o*=2,k*=2)
            for(int i=0;i<n;i+=o)
                for(int j=0;j<k;j++)
                    f[i+j]=(f[i+j+k]*x+f[i+j])%mod;
    }
    void OR(int x){
        for(int o=2,k=1;o<=n;o*=2,k*=2)
            for(int i=0;i<n;i+=o)
                for(int j=0;j<k;j++)
                    f[i+j+k]=(f[i+j+k]+f[i+j]*x)%mod;
    }
    void XOR(int x){
        for(int o=2,k=1;o<=n;o*=2,k*=2)
            for(int i=0;i<n;i+=o)
                for(int j=0;j<k;j++){
                    int y=f[i+j+k];
                    f[i+j+k]=(f[i+j]-f[i+j+k]+mod)%mod;
```

```
                    f[i+j]=(f[i+j]+y)%mod;
                    f[i+j]=f[i+j]*x%mod;f[i+j+k]=f[i+j+k]*x%mod;
                }
    }
    void print(){
        for(int i=0;i<n;i++)printf("%d ",f[i]);puts("");
    }
}A,B;
void init(){for(int i=0;i<n;i++)A.f[i]=a[i],B.f[i]=b[i];}
void get(){for(int i=0;i<n;i++){A.f[i]=A.f[i]*B.f[i]%mod;}}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    n=read();n=(1<<n);
    for(int i=0;i<n;i++)a[i]=read();
    for(int i=0;i<n;i++)b[i]=read();
    init();A.OR(1),B.OR(1),get(),A.OR(mod-1);A.print();
    init();A.AND(1),B.AND(1),get(),A.AND(mod-1);A.print();
    init();A.XOR(1),B.XOR(1),get(),A.XOR(mod/2+1);A.print();
    return 0;
}
```

5 其他算法

5.1 线性基

```
long long pos[105],n,a[105];
int main(){
    cin>>n;
    for(int i=1;i<=n;i++)cin>>a[i];
    for(int i=1;i<=n;i++){
        for(int j=52;j>=0;j--){
            if(!(a[i]>>j)) continue;
            if(!pos[j]) {pos[j]=a[i];break;}
            a[i]^=pos[j];
        }
    }
    long long ans=0;
    for(int i=54;i>=0;i--){
        ans=max(ans,ans^pos[i]);
    }
    cout<<ans;
    return 0;
}
```

5.2 Prufer 序列

Prufer 序列可以将一个带标号 n个节点的树用 [1,n] 中的 n−2个整数表示，即 n 个点的完全图的生成树与长度为 n−2 值域为 [1,n] 的数列构成的双射。

Prufer 序列可以方便的解决一类树相关的计数问题，比如凯莱定理：n个点的完全图的生成树有 n^{n-2} 个。

```
int make_prufer()
{
    for(int i=1;i<n;i++)f[i]=read(),in[f[i]]++,in[i]++;
    int t=1,i=0,x,ans=0;
    while(i<n-1)
    {
        while(in[t]!=1)t++;
        p[++i]=f[t];x=f[t],in[f[t]]--,in[t]--;
        while(in[x]==1&&x<t&&i<n-1)p[++i]=f[x],in[f[x]]--,in[x]--,x=f[x];
    }
    for(int i=1;i<=n-2;i++)ans=ans^(i*p[i]);
    return ans;
}
int make_fa()
{
    for(int i=1;i<n-1;i++)p[i]=read(),in[p[i]]++;
    for(int i=1;i<=n;i++)in[i]++;
    int t=1,i=0,x,ans=0;
    while(i<n-2)
    {
        while(in[t]!=1)t++;
        f[t]=p[++i];in[p[i]]--;in[t]--;x=p[i];//cout<<t<<" "<<p[i]<<" "<<i<<" "
<<f[t]<<endl;cout<<x<<" "<<i<<endl;
        while(in[x]==1&&x<t&&i<n-2)f[x]=p[++i],in[p[i]]--,in[x]--,x=p[i];
    }
    for(int i=1;i<n;i++)f[i]==0?f[i]=n:f[i]=f[i],ans=ans^(i*f[i]);
    return ans;

}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    n=read();opt=read();
    if(opt==1)cout<<make_prufer();
    else cout<<make_fa();
    return 0;
}
```

5.3 最小斯坦纳树

给定一个包含 n 个结点和 m 条带权边的无向连通图 $G=(V,E)$。

再给定包含 k 个结点的点集 S，选出 G 的子图 $G'=(V',E')$，使得：

1. $S\subseteq V'$；
2. G' 为连通图；
3. E' 中所有边的权值和最小。

你只需要求出 E' 中所有边的权值和。

```
int n,m,x,y,z,f[105][maxn],a[maxn],k,vis[105];
vector<pi>e[maxn];
```

```
priority_queue<pi>q;
void dij(int s)
{
    memset(vis,0,sizeof vis);
    while(!q.empty())
    {
        int x=q.top().se;q.pop();//cout<<x<<endl;
        if(vis[x])continue;vis[x]=1;
        for(auto i:e[x])
            if(f[x][s]+i.se<f[i.fi][s])
                f[i.fi][s]=f[x][s]+i.se,q.push(mp(-f[i.fi][s],i.fi));
    }
}
signed main()
{
    //freopen(".in","r",stdin);
    //freopen(".out","w",stdout);
    n=read(),m=read();k=read();memset(f,0x3f,sizeof f);
    for(int
i=1;i<=m;i++)x=read(),y=read(),z=read(),e[x].push_back(mp(y,z)),e[y].push_back(mp
(x,z));
    for(int i=1;i<=k;i++)a[i]=read(),f[a[i]][(1<<(i-1))]=0;
    for(int j=1;j<(1<<k);j++)
    {
        for(int i=1;i<=n;i++)
        {
            for(int s=j&(j-1);s;s=j&(s-1))
                f[i][j]=min(f[i][j],f[i][s]+f[i][j^s]);
            if(f[i][j]!=0x3f3f3f3f)q.push(mp(-f[i][j],i));
        }
        dij(j);
        //for(int i=1;i<=n;i++)cout<<f[i][j]<<" ";cout<<endl;
    }
    cout<<f[a[1]][(1<<k)-1];
    return 0;
}
```