AC 自动机

```
struct ACautomaton
{
    struct Node
    {
        int val,fail,son[26];
    }t[MAXN];
    int cnt;
    void insert_Trie(char *s,int val)
    {
        int u=0,n=strlen(s);
        for(int i=0;i<n;i++)
        {
            if(!t[u].son[s[i]-'a'])t[u].son[s[i]-'a']=++cnt;
            u=t[u].son[s[i]-'a'];
        }
        t[u].val+=val;
    }
    queue<int> q;
    void getfail()
    {
        for(int i=0;i<26;i++)if(t[0].son[i])q.push(t[0].son[i]);
        while(!q.empty())
        {
            int u=q.front();q.pop();
            for(int i=0;i<26;i++)
            {
                if(t[u].son[i])
                {
                    t[t[u].son[i]].fail=t[t[u].fail].son[i];
                    q.push(t[u].son[i]);
                }
                else t[u].son[i]=t[t[u].fail].son[i];
            }
        }
    }
}G;
```

带花树

```
int Head[MAXN], Next[MAXN], Go[MAXN], Pre[1010], Nxt[1010], F[1010], S[1010],
Q[1010], Vis[1010], *Top = Q, Cnt = 0, Tim = 0, x, y;

inline void addedge(int x, int y)
{
    Go[++Cnt] = y;
    Next[Cnt] = Head[x];
    Head[x] = Cnt;
    Go[++Cnt] = x;
    Next[Cnt] = Head[y];
    Head[y] = Cnt;
}
```

```
int find(int x)
{
    return x == F[x] ? x : F[x] = find(F[x]);
}

int lca(int x, int y)
{
    for(Tim++, x = find(x), y = find(y); ; x ^= y ^= x ^= y)
        if(x)
        {
            if(Vis[x] == Tim) return x;
            Vis[x] = Tim;
            x = find(Pre[Nxt[x]]);
        }
}

void blossom(int x, int y, int l)
{
    while(find(x) != l)
    {
        Pre[x] = y, y = Nxt[x];
        S[*Top = y] = 0, *Top++;
        if(F[x] == x) F[x] = l;
        if(F[y] == y) F[y] = l;
        x = Pre[y];
    }
    return ;
}

int Match(int x,int n)
{
    for(int i = 1; i <= n; i++) F[i] = i;
    memset(S, -1, sizeof S);
    S[*(Top = Q) = x] = 0, Top++;
    for(int *i = Q; i != Top; *i++)
        for(int T = Head[*i]; T; T = Next[T])
        {
            int g = Go[T];
            if(S[g] == -1)
            {
                Pre[g] = *i, S[g] = 1;
                if(!Nxt[g])
                {
                    for(int u = g, v = *i, lst; v; u = lst, v = Pre[u])
                        lst = Nxt[v], Nxt[v] = u, Nxt[u] = v;
                    return 1;
                }
                S[*Top = Nxt[g]] = 0, Top++;
            }
            else if(!S[g] && find(g) != find(*i))
            {
                int l = lca(g, *i);
                blossom(g, *i, l);
                blossom(*i, g, l);
            }
        }
    return 0;
}
```

```
int d[1000],idx[1000][2],n,m;

void init()
{
    Cnt=Tim=0;
    memset(Nxt,0,sizeof Nxt);
    memset(Head,0,sizeof Head);
    memset(Pre,0,sizeof Pre);
    memset(Q,0,sizeof Q);
    memset(Vis,0,sizeof Vis);
    memset(idx,0,sizeof idx);
    Top=Q;
}

for(int i=n;i;i--) if (!Nxt[i]) ans+=Match(i,n);
W(ans);
for(int i=1;i<=n;i++)printf("%d ",Nxt[i]);
```

DLX

```
struct Dancing_links
{
    int n,m,cnt;
    int l[MAXN],r[MAXN],u[MAXN],d[MAXN],col[MAXN],row[MAXN];
    int h[MAXN],s[MAXN],ansk[MAXN];
    void init(int m)
    {
        for(int i=0;i<=m;i++)
        {
            r[i]=i+1;
            l[i]=i-1;
            u[i]=d[i]=i;
        }
        r[m]=0;
        l[0]=m;
        memset(h,-1,sizeof(h));
        memset(s,0,sizeof(s));
        cnt=m;
    }
    void link(int R,int C)
    {
        cnt++;
        s[C]++;
        row[cnt]=R;
        col[cnt]=C;
        u[cnt]=C;
        d[cnt]=d[C];
        u[d[C]]=cnt;
        d[C]=cnt;
        if(h[R]==-1)h[R]=r[cnt]=l[cnt]=cnt;
        else
        {
            r[cnt]=h[R];
            l[cnt]=l[h[R]];
            r[l[h[R]]]=cnt;
            l[h[R]]=cnt;
        }
    }
    void remove(int c)
    {
        r[l[c]]=r[c];
        l[r[c]]=l[c];
        for(int i=d[c];i!=c;i=d[i])
            for(int j=r[i];j!=i;j=r[j])
            {
                u[d[j]]=u[j];
                d[u[j]]=d[j];
                s[col[j]]--;
            }
    }
    void recover(int c)
    {
        for(int i=u[c];i!=c;i=u[i])
            for(int j=l[i];j!=i;j=l[j])
            {
                u[d[j]]=j;
                d[u[j]]=j;
                s[col[j]]++;
            }
        r[l[c]]=c;
        l[r[c]]=c;
    }
    bool dance(int dep)
    {
        if(!r[0])
        {
            for(int i=0;i<dep;i++)printf("%d ",ansk[i]);
            return true;
        }
        int c=r[0];for(int i=r[0];i;i=r[i])if(s[i]<s[c])c=i;
        remove(c);
        for(int i=d[c];i!=c;i=d[i])
        {
            ansk[dep]=row[i];
            for(int j=r[i];j!=i;j=r[j])remove(col[j]);
            if(dance(dep+1))return true;
            for(int j=l[i];j!=i;j=l[j])recover(col[j]);
        }
        recover(c);
        return false;
    }
}D;
//先按列数init，输入图后如果(i,j)为1则link(i,j)，最后dance(0)出结果。
```

exSAM

```
struct exSAM
{
    int len[MAXN],fail[MAXN],val[MAXN],ch[MAXN][30];
    int sz;
    void init()
    {
        sz=len[0]=(fail[0]=-1)+1;sz++;
        memset(ch[0],0,sizeof(ch[0]));
    }
    int extend(char c,int last,int id)
    {
        if(ch[last][c-'a'])
        {
            int p=last,u=ch[last][c-'a'];
            if(len[p]+1==len[u])return u;
            else
            {
                int clone=++sz;len[clone]=len[p]+1;
                for(int i=0;i<26;i++)ch[clone][i]=ch[u][i];
                while(p&&ch[p][c-'a']==u)ch[p][c-'a']=clone,p=fail[p];
                fail[clone]=fail[u];fail[u]=clone;
                return clone;
            }
        }
        int cur=++sz;
        memset(ch[cur],0,sizeof(ch[cur]));
        len[cur]=len[last]+1;
        int p=last;
        while(p&&!ch[p][c-'a'])
        {
            ch[p][c-'a']=cur;
            p=fail[p];
        }
        if(p)
        {
            int q=ch[p][c-'a'];
            if(len[p]+1==len[q])fail[cur]=q;
            else
            {
                int clone=++sz;
                fail[clone]=fail[q];
                for(int i=0;i<26;i++)ch[clone][i]=ch[q][i];
                len[clone]=len[p]+1;
                val[clone]=0;
                while(p&&ch[p][c-'a']==q)
                {
                    ch[p][c-'a']=clone;
                    p=fail[p];
                }
                fail[q]=fail[cur]=clone;
            }
        }
        else fail[cur]=1;
        return cur;
    }
    int ord[MAXN],cnt[MAXN];
    void _sort()
    {
        memset(cnt,0,(sz+5)*sizeof(int));
        for(int i=1;i<=sz;i++)cnt[len[i]]++;
        for(int i=1;i<=sz;i++)cnt[i]+=cnt[i-1];
        for(int i=sz;i>=1;i--)ord[cnt[len[i]]--]=i;
    }
}G;
```

FFT

```
#define MN 262144
const double pi=acos(-1);
struct cp
{
    double r,i;
    cp(double r=0,double i=0):r(r),i(i){}
    cp operator+(cp b){return cp(r+b.r,i+b.i);}
    cp operator-(cp b){return cp(r-b.r,i-b.i);}
    cp operator*(cp b){return cp(r*b.r-i*b.i,r*b.i+i*b.r);}
}w[2][MN+5],F[MN+5],G[MN+5];
int N,R[MN+5],u[MN+5][6][6],n,m;
void init(int n)
{
    for(N=1;N<=n;N<<=1);
    cp g(cos(2*pi/N),sin(2*pi/N));int i,j,k;
    for(i=w[0][0].r=1;i<N;++i)w[0][i]=w[0][i-1]*g;
    for(i=w[1][0].r=1;i<N;++i)w[1][i]=w[0][N-i];
    for(i=j=0;i<N;R[++i]=j)for(k=N>>1;(j^=k)<k;k>>=1);
}
void fft(cp*x,int v)
{
    int i,j,k;
    for(i=0;i<N;++i)if(i<R[i])swap(x[i],x[R[i]]);
    for(i=1;i<N;i<<=1)for(j=0;j<N;j+=i<<1)for(k=0;k<i;++k)
    {
        cp p=x[i+j+k]*w[v][N/(i<<1)*k];
        x[i+j+k]=x[j+k]-p;x[j+k]=x[j+k]+p;
    }
    if(v)for(i=0;i<N;++i)x[i].r/=N,x[i].i/=N;
}

int main()
{
    init(n+m);
    fft(F,0),fft(G,0);
    for(int i=0;i<N;i++)F[i]=F[i]*G[i];
    fft(F,1);
    for(int i=0;i<n+m-1;i++)printf("%d",(int)(F[i].r+0.5));
    return 0;
}
```

FST

```
#include<bits/stdc++.h>
using namespace std;
#define fi first
#define se second
#define SZ(x) ((int)x.size())
#define ALL(x) (x.begin(),x.end())
typedef long long ll;
typedef pair<int,int> pii;
typedef double db;
const int MAXN=6e5+10,MAXM=3e6+10;
const int INF=INT_MAX,SINF=0x3f3f3f3f;
const ll llINF=LLONG_MAX;
const int MOD=1e9+9,mod=998244353;

int n;
int a[22][MAXM],b[22][MAXM],h[22][MAXM];

int add(int a,int b)
{
    return a+b>=MOD?a+b-MOD:a+b;
}

int sub (int a,int b)
{
    return a<b?a-b+MOD:a-b;
}

int mul (int a,int b)
{
    return ((ll)a*b%MOD+MOD)%MOD;
}

void OR(int *f,int x)
{
     for (int o = 2, k = 1; o <= n; o <<= 1, k <<= 1)
        for (int i = 0; i < n; i += o)
            for (int j = 0; j < k; j++)
                f[i+j+k] = add(f[i+j+k],mul(f[i+j],x));
}

int B(int x)
{
    return __builtin_popcount(x);
}

int main()
{
    scanf("%d",&n);
    int lim=n;
    n=(1<<n);
    for(int i=0;i<n;i++)scanf("%d",&a[B(i)][i]);
    for(int i=0;i<n;i++)scanf("%d",&b[B(i)][i]);
    for(int i=0;i<=lim;i++)
    {
        OR(a[i],1);
        OR(b[i],1);
    }
    for(int i=0;i<=lim;i++)
        for(int j=0;j<=i;j++)
            for(int k=0;k<n;k++)h[i][k]=add(h[i][k],mul(a[j][k],b[i-j][k]));
    for(int i=0;i<=lim;i++)OR(h[i],-1);
    for(int i=0;i<n;i++)printf("%d ",h[B(i)][i]);
    return 0;
}
```

FWT

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

```cpp
#define fi first
#define se second
#define SZ(x) ((int)x.size())
#define ALL(x) (x.begin(),x.end())
typedef long long ll;
typedef pair<int,int> pii;
typedef double db;
const int MAXN=6e5+10,MAXM=4e6+10;
const int INF=INT_MAX,SINF=0x3f3f3f3f;
const ll llINF=LLONG_MAX;
const int MOD=998244353,mod=998244353;
const int Lim=1<<21;

int n;
int a[MAXN],b[MAXN],A[MAXN],B[MAXN];

int add(int a,int b)
{
    return a+b>=MOD?a+b-MOD:a+b;
}

int sub (int a,int b)
{
    return ((ll)a*b%MOD+MOD)%MOD;
}

int mul (int a,int b)
{
    return (ll)a*b%MOD;
}

void OR(int *f,int x)
{
     for (int o = 2, k = 1; o <= n; o <<= 1, k <<= 1)
        for (int i = 0; i < n; i += o)
            for (int j = 0; j < k; j++)
                f[i+j+k] = add(f[i+j+k],mul(f[i+j],x));
}

void AND(int *f,int x)
{
    for (int o = 2, k = 1; o <= n; o <<= 1, k <<= 1)
        for (int i = 0; i < n; i += o)
            for (int j = 0; j < k; j++)
                f[i+j] = add(f[i+j],mul(f[i+j+k],x));
}

void XOR(int *f,int p)
{
    int inv2=(MOD+1)/2;
    for (int o = 2, k = 1; o <= n; o <<= 1, k <<= 1)
        for (int i = 0; i < n; i += o)
            for (int j = 0; j < k; j++)
            {
                int u,t;
                u=f[j+i];t=f[j+k+i];
                f[j+i]=mul(p^1?inv2:1,add(u,t));
                f[j+k+i]=mul(p^1?inv2:1,sub(u,t));
            }
}

int main()
{
    scanf("%d",&n);
    n=(1<<n);
    for(int i=0;i<n;i++)scanf("%d",&a[i]);
    for(int i=0;i<n;i++)scanf("%d",&b[i]);
    copy(a,a+n,A);copy(b,b+n,B);
    OR(A,1);OR(B,1);for(int i=0;i<n;i++)A[i]=mul(A[i],B[i]);OR(A,MOD-1);
    for(int i=0;i<n;i++)printf("%d ",A[i]);puts("");
    copy(a,a+n,A);copy(b,b+n,B);
    AND(A,1);AND(B,1);for(int i=0;i<n;i++)A[i]=mul(A[i],B[i]);AND(A,MOD-1);
    for(int i=0;i<n;i++)printf("%d ",A[i]);puts("");
    copy(a,a+n,A);copy(b,b+n,B);
    XOR(A,1);XOR(B,1);for(int i=0;i<n;i++)A[i]=mul(A[i],B[i]);XOR(A,0);
    for(int i=0;i<n;i++)printf("%d ",A[i]);puts("");
    return 0;
}
```

HK

```
struct HK
{
    vector<int> g[MAXN]; int n1, n2;
    int lnk[MAXN], dis[MAXN], dm; bool vis[MAXN];

    void clr_hk(int n, int m) {
        n1 = n; n2 = m;
        for (int i = 1; i <= n1 + n2; ++i)
            g[i].clear();
    }

    bool bfs_hk() {
        queue<int> q; dm = INT_MAX;
        fill(dis + 1, dis + n1 + n2 + 1, -1);
        for (int i = 1; i <= n1; ++i)
            if (!lnk[i]) q.push(i), dis[i] = 0;
        while (!q.empty()) {
            int u = q.front(); q.pop();
            if (dis[u] > dm) break;
            for (int v : g[u]) {
                if (dis[v] != -1) continue;
                dis[v] = dis[u] + 1;
                if (!lnk[v]) dm = dis[v];
                else dis[lnk[v]] = dis[v] + 1, q.push(lnk[v]);
            }
        }
        return dm != INT_MAX;
    }

    int dfs_hk(int u) {
        for (int v : g[u]) {
            if (vis[v] || dis[v] != dis[u] + 1) continue;
            vis[v] = 1;
            if (lnk[v] && dis[v] == dm) continue;
            if (lnk[v] && !dfs_hk(lnk[v])) continue;
            lnk[v] = u; lnk[u] = v; return 1;
        }
        return 0;
    }

    int hk() {
        fill (lnk + 1, lnk + n1 + n2 + 1, 0);
        int res = 0;
        while (bfs_hk()) {
            fill (vis + 1, vis + n1 + n2 + 1, 0);
            for (int i = 1; i <= n1; ++i)
                if (!lnk[i] && dfs_hk(i)) res++;
        }
        return res;
    }

    int n,m;
    inline void add(int u,int v)
    {
        g[u].push_back(v+n);
    }
}G;
```

KM

```
int n,m;
ll lx[MAXN],ly[MAXN],slack[MAXN],g[MAXN][MAXN];
int px[MAXN],py[MAXN],pre[MAXN];
bool vx[MAXN],vy[MAXN];

void aug(int u)
{
    int v=u;
    while(u)
    {
        v=px[pre[u]];
        px[pre[u]]=u;
        py[u]=pre[u];
        u=v;
    }
}
void bfs(int s)
{
    memset(vx,0,sizeof(vx));
    memset(vy,0,sizeof(vy));
    memset(slack,0x3f,sizeof(slack));
    queue<int> q;
    q.push(s);
    while(1)
    {
        while(!q.empty())
        {
            int u=q.front();q.pop();
            vx[u]=1;
            for(int i=1;i<=n;i++)if(!vy[i]&&lx[u]+ly[i]-g[u][i]<slack[i])
            {
                slack[i]=lx[u]+ly[i]-g[u][i];
                pre[i]=u;
                if(!slack[i])
                {
                    vy[i]=1;
                    if(!py[i])return aug(i);
                    else q.push(py[i]);
                }
            }
        }
        ll d=llINF;
        for(int i=1;i<=n;i++)if(!vy[i])d=min(d,slack[i]);
        for(int i=1;i<=n;i++)
        {
            if(vx[i])lx[i]-=d;
            if(vy[i])ly[i]+=d;
            else slack[i]-=d;
        }
        for(int i=1;i<=n;i++)if(!vy[i]&&!slack[i])
        {
            vy[i]=1;
            if(!py[i])return aug(i);
            else q.push(py[i]);
        }
    }
}

void init(int n)
{
    for(int i=1;i<=n;i++)
        for(int j=1;j<=n;j++)
        g[i][j]=-INF;
    for(int i=1;i<=n;i++)lx[i]=-INF;
}

ll KM(int n)
{
    ll ans=0;
    for(int i=1;i<=n;i++)
        for(int j=1;j<=n;j++)
        lx[i]=max(lx[i],g[i][j]);
    for(int i=1;i<=n;i++)bfs(i);
    for(int i=1;i<=n;i++)ans+=lx[i]+ly[i];
    return ans;
}

// init at first, the function KM is the answer.
// Remember to use max in add_edge to avoid multi edge.
```

KMP

```
void kmp(string &a,string &b)
{
    string tmp;
    tmp+=a;
    tmp+='@';
    tmp+=b;
    for(int i=1;i<SZ(tmp);i++)
    {
        int u=nxt[i-1];
        while(u>0&&tmp[u]!=tmp[i])u=nxt[u-1];
        if(tmp[u]==tmp[i])u++;
        nxt[i]=u;
    }
}
```

LCT

```
namespace LCT
{
    int col[MAXN],f[MAXN],ch[MAXN][2],laz[MAXN];
    int lch(int x){return ch[x][0];}
    int rch(int x){return ch[x][1];}
    bool isroot(int x){return lch(f[x])!=x&&rch(f[x])!=x;}
    void cover(int x,int k){col[x]=laz[x]=k;}
    bool isrch(int x){return rch(f[x])==x;}
    void pushdown(int x)
    {
        if(laz[x])
        {
            if(lch(x))cover(lch(x),laz[x]);
            if(rch(x))cover(rch(x),laz[x]);
            laz[x]=0;
        }
    }
    void rotate(int x)
    {
        int y=f[x],z=f[y];
        bool k=isrch(x);
        int w=ch[x][!k];
        if(!isroot(y))ch[z][isrch(y)]=x;
        f[x]=z;
        ch[y][k]=w;
        if(w)f[w]=y;
        ch[x][!k]=y;
        f[y]=x;
    }
    void pushdownall(int x)
    {
        if(!isroot(x))pushdownall(f[x]);
        pushdown(x);
    }
    void splay(int x)
    {
        pushdownall(x);
        while(!isroot(x))
        {
            if(!isroot(f[x]))rotate(isrch(x)^isrch(f[x])?x:f[x]);
            rotate(x);
        }
    }
    void access(int p)
    {
        int x=G.pos[p],y=0;
        for(;x;y=x,x=f[x])
        {
            splay(x);
            if(int k=col[x])
                S.change(1,k-G.len[x]+1,k-G.len[f[x]],1,n,-1);
            ch[x][1]=y;
        }
        cover(y,p);
        S.change(1,1,p,1,n,1);
    }
    void build()
    {
        for(int i=1;i<=G.sz;i++)
            f[i]=G.fail[i];
    }
}
```

Manacher

```
int p[MAXN];
void manacher(char *x){
    int id,mx=0,i;
    for(i=1;i<=(n<<1)+1;i++){
        if(i<mx)p[i]=min(p[2*id-i],mx-i);
        else p[i]=1;
        while(x[i-p[i]]==x[i+p[i]]&&i-p[i]>=1&&i+p[i]<=(n<<1)+1) p[i]++;
        if(mx<i+p[i])mx=p[id=i]+i;
    }
}

t[++cnt]='#';
for(int i=1;i<=n;i++)
{
    t[++cnt]=s[i];
    t[++cnt]='#';
}
manacher(t);
```

NTT

```
int add (int a, int b) { return (a += b) >= mod ? a - mod : a; }
int sub (int a, int b) { return (a -= b) >= 0 ? a : a + mod; }
int mul (long long a, int b) { return a * b % mod; }
int ksm (int a, int b) { int ret = 1; while (b) { if (b & 1) ret = 1ll * ret * a
% mod; a = 1ll * a * a % mod; b >>= 1; } return ret; }

int rev[Lim+5],lg2[Lim+5],rt[Lim+5],irt[Lim+5];
int n,k,lim,type;

void init()         //求原根
{
    for(int i=2;i<=Lim;i++)lg2[i]=lg2[i>>1]+1;
    rt[0]=1;
    rt[1]=ksm(3,(MOD-1)/Lim);   //第一个原根
    irt[0]=1;
    irt[1]=ksm(rt[1],MOD-2);   //费马小
    for(int i=2;i<=Lim;i++)
    {
        rt[i]=mul(rt[i-1],rt[1]);
        irt[i]=mul(irt[i-1],irt[1]);
    }
}

void NTT(Vi &f,int type,int lim)
{
    f.resize(lim);
    int w,y,l=lg2[lim]-1;
    for(int i=1;i<lim;i++)
    {
        rev[i]=(rev[i>>1]>>1)|((i&1)<<l);
        if(i>=rev[i])continue;
        swap(f[i],f[rev[i]]);           //蝴蝶
    }
    l=Lim>>1;
    for(int mid=1;mid<lim;mid<<=1)
    {
        for(int j=0;j<lim;j+=(mid<<1))
        {
            for(int k=0;k<mid;k++)
            {
                w=type==1?rt[l*k]:irt[l*k];
                y=mul(w,f[j|k|mid]);
                f[j|k|mid]=sub(f[j|k],y);
                f[j|k]=add(f[j|k],y);
            }
        }
        l>>=1;
    }
    if(type==1)return;
    y=ksm(lim,MOD-2);
    for(int i=0;i<lim;i++)
        f[i]=mul(f[i],y);
}

void NTTTMD(Vi &F,Vi &G)
{
    int n=SZ(F)+SZ(G);
    lim=1;
    while(lim<=n)lim<<=1;F.resize(lim);G.resize(lim);
    NTT(F,1,lim),NTT(G,1,lim);
    for(int i=0;i<lim;i++)F[i]=mul(F[i],G[i]);
    NTT(F,-1,lim);
    F.resize(n-1);
}
```

$O(1)$ 求 gcd

```
int a[MAXN],b[MAXN],n;
int ans,k[MAXM][3],_gcd[1005][1005];

int gcd(int a,int b)
{
    int g=1;
    for(int i=0;i<3;i++)
    {
        int tmp;
        if(k[a][i]>1000)
        {
            if(b%k[a][i])tmp=1;
            else tmp=k[a][i];
        }
        else tmp=_gcd[k[a][i]][b%k[a][i]];
        b/=tmp;
        g*=tmp;
    }
    return g;
}

bool isprime[MAXM];
int p[MAXN],pcnt;

void Linear_shaker()
{
    k[1][0]=k[1][1]=k[1][2]=1;
    for(int i=2;i<=1e6;i++)
    {
        if(!isprime[i])p[++pcnt]=i,k[i][0]=k[i][1]=1,k[i][2]=i;
        for(int j=1;p[j]*i<=1e6;j++)
        {
            isprime[i*p[j]]=1;
            k[i*p[j]][0]=k[i][0]*p[j];
            k[i*p[j]][1]=k[i][1];
            k[i*p[j]][2]=k[i][2];
            if(k[i*p[j]][0]>k[i*p[j]][1])swap(k[i*p[j]][1],k[i*p[j]][0]);
            if(k[i*p[j]][1]>k[i*p[j]][2])swap(k[i*p[j]][2],k[i*p[j]][1]);
            sort(k[i*p[j]],k[i*p[j]]+3);
            if(i%p[j]==0)break;
        }
    }
    for(int i=1;i<=1e3;i++)_gcd[i][0]=_gcd[0][i]=i;
    for(int i=1;i<=1e3;i++)
        for(int j=1;j<=i;j++)
        _gcd[i][j]=_gcd[j][i]=_gcd[i%j][j];
}

int main()
{
    Linear_shaker();
    return 0;
}
```

Prufer 序列与树的转化

```
void f_to_P()
{
    for(register int i=1;i<n;i++)read(f[i]),du[f[i]]++;
    for(register int i=1,j=1;i<=n-2;i++,j++)
    {
        while(du[j])j++;
        p[i]=f[j];
        while(i<=n-2&&!--du[p[i]]&&p[i]<j)p[i+1]=f[p[i]],i++;
    }
}

void p_to_f()
{
    for(register int i=1;i<=n-2;i++)read(p[i]),du[p[i]]++;
    p[n-1]=n;
    for(register int i=1,j=1;i<n;i++,j++)
    {
        while(du[j])j++;f[j]=p[i];
        while(i<n&&!--du[p[i]]&&p[i]<j)f[p[i]]=p[i+1],i++;
    }
}
```

SA

```
int cnt[MAXN],sa[MAXN],rk[MAXN],px[MAXN],height[MAXN],id[MAXN],oldrk[MAXN],p;
bool cmp(int x, int y, int w) {
  return oldrk[x] == oldrk[y] && oldrk[x + w] == oldrk[y + w];
}
void get_SA()
{
    for (int i = 1; i <= n; ++i) ++cnt[rk[i] = s[i]];
    for (int i = 1; i <= m; ++i) cnt[i] += cnt[i - 1];
    for (int i = n; i >= 1; --i) sa[cnt[rk[i]]--] = i;

    for (int w = 1; w < n; w <<= 1, m = p,p=0) {  // m=p 就是优化计数排序值域
    for (int i = n; i > n - w; --i) id[++p] = i;
    for (int i = 1; i <= n; ++i)
      if (sa[i] > w) id[++p] = sa[i] - w;
    memset(cnt, 0, sizeof(cnt));
    for (int i = 1; i <= n; ++i) ++cnt[px[i] = rk[id[i]]];
    for (int i = 1; i <= m; ++i) cnt[i] += cnt[i - 1];
    for (int i = n; i >= 1; --i) sa[cnt[px[i]]--] = id[i];
    memcpy(oldrk, rk, sizeof(rk));
    for (int p = 0, i = 1; i <= n; ++i)
      rk[sa[i]] = cmp(sa[i], sa[i - 1], w) ? p : ++p;
  }
}
void get_height()
{
    for (int i = 1, k = 0; i <= n; ++i)
    {
        if (k) --k;
        while (s[i + k] == s[sa[rk[i] - 1] + k]) ++k;
        height[rk[i]] = k;
    }
}
```

SAIS

```
template<size_t size>
struct SuffixArray {
    bool t[size << 1];
    int sa[size], ht[size], rk[size];
    inline bool islms(const int i, const bool *t) {
        return i > 0 && t[i] && !t[i - 1];
    }
    template<class T>
    inline void sort(T s, int *sa, const int len, const int sigma, const int sz,
bool *t, int *b, int *cb, int *p) {
        memset(b, 0, sizeof(int) * sigma);
        memset(sa, -1, sizeof(int) * len);
        for (register int i = 0; i < len; i++) b[s[i]]++;
        cb[0] = b[0];
        for (register int i = 1; i < sigma; i++) cb[i] = cb[i - 1] + b[i];
        for (register int i = sz - 1; i >= 0; i--) sa[--cb[s[p[i]]]] = p[i];
        for (register int i = 1; i < sigma; i++) cb[i] = cb[i - 1] + b[i - 1];
        for (register int i = 0; i < len; i++) if (sa[i] > 0 && !t[sa[i] - 1])
sa[cb[s[sa[i] - 1]]++] = sa[i] - 1;
        cb[0] = b[0];
        for (register int i = 1; i < sigma; i++) cb[i] = cb[i - 1] + b[i];
        for (register int i = len - 1; i >= 0; i--) if (sa[i] > 0 && t[sa[i] -
1]) sa[--cb[s[sa[i] - 1]]] = sa[i] - 1;
    }
    template<class T>
    inline void sais(T s, int *sa, const int len, bool *t, int *b, int *b1,
const int sigma) {
        register int i, j, x, p = -1, cnt = 0, sz = 0, *cb = b + sigma;
        for (t[len - 1] = 1, i = len - 2; i >= 0; i--) t[i] = s[i] < s[i + 1] ||
(s[i] == s[i + 1] && t[i + 1]);
        for (i = 1; i < len; i++) if (t[i] && !t[i - 1]) b1[sz++] = i;
        sort(s, sa, len, sigma, sz, t, b, cb, b1);
        for (i = sz = 0; i < len; i++) if (islms(sa[i], t)) sa[sz++] = sa[i];
        for (i = sz; i < len; i++) sa[i] = -1;
        for (i = 0; i < sz; i++) {
            for (x = sa[i], j = 0; j < len; j++) {
                if (p == -1 || s[x + j] != s[p + j] || t[x + j] != t[p + j]) {
cnt++, p = x; break; }
                else if (j > 0 && (islms(x + j, t) || islms(p + j, t))) break;
            }
            sa[sz + (x >>= 1)] = cnt - 1;
        }
        for (i = j = len - 1; i >= sz; i--) if (sa[i] >= 0) sa[j--] = sa[i];
        register int *s1 = sa + len - sz, *b2 = b1 + sz;
        if (cnt < sz) sais(s1, sa, sz, t + len, b, b1 + sz, cnt);
        else for (i = 0; i < sz; i++) sa[s1[i]] = i;
        for (i = 0; i < sz; i++) b2[i] = b1[sa[i]];
        sort(s, sa, len, sigma, sz, t, b, cb, b2);
    }
    template<class T>
    inline void getHeight(T s, int n) {
        for (register int i = 1; i <= n; i++) rk[sa[i]] = i;
        register int j = 0, k = 0;
        for (register int i = 0; i < n; ht[rk[i++]] = k)
            for (k ? k-- : 0, j = sa[rk[i] - 1]; s[i + k] == s[j + k]; k++);
    }
    template<class T>
    inline void init(T s, const int len, const int sigma) {
        sais(s, sa, len+1, t, rk, ht, sigma), rk[0] = 0;
    }
    int lg[size],st[size][20],n;
    void getlcp(int sz)
    {
        n=sz;
        for(int i=1;i<=n;i++)lg[i]=lg[i-1]+(1<<lg[i-1]+1==i);
        for(int i=1;i<=n;i++)st[i][0]=ht[i];
        for(int j=1;1<<j<=n;j++)
            for(int i=1;i+(1<<j)-1<=n;i++)
            st[i][j]=min(st[i][j-1],st[i+(1<<(j-1))][j-1]);
    }
    int lcp(int i,int j)
    {
        if(i==j)return n-i+1;
        i=rk[i];j=rk[j];
        if(i>j)swap(i,j);
        i++;int k=lg[j-i+1];
        //W(i,j,k,j-(1<<k)+1,min(st[i][k],st[j-(1<<k)+1][k]));
        return min(st[i][k],st[j-(1<<k)+1][k]);
    }
};
SuffixArray<MAXN>SA;
```

SAM

```
struct SAM
{
    int len[MAXN],fail[MAXN],val[MAXN],ch[MAXN][30];
    int sz,last;
    void init()
    {
        sz=last=len[0]=(fail[0]=-1)+1;
        memset(ch[0],0,sizeof(ch[0]));
    }
    void extend(char c)
    {
        int cur=++sz;
        memset(ch[cur],0,sizeof(ch[cur]));
        len[cur]=len[last]+1;
        int p=last;
        while(~p&&!ch[p][c-'a'])
        {
            ch[p][c-'a']=cur;
            p=fail[p];
        }
        if(~p)
        {
            int q=ch[p][c-'a'];
            if(len[p]+1==len[q])fail[cur]=q;
            else
            {
                int clone=++sz;
                fail[clone]=fail[q];
                for(int i=0;i<26;i++)ch[clone][i]=ch[q][i];
                len[clone]=len[p]+1;
                val[clone]=0;
                while(~p&&ch[p][c-'a']==q)
                {
                    ch[p][c-'a']=clone;
                    p=fail[p];
                }
                fail[q]=fail[cur]=clone;
            }
        }
        else fail[cur]=0;
        last=cur;
        val[cur]=1;
    }
    int ord[MAXN],cnt[MAXN];
    void _sort()
    {
        memset(cnt,0,(sz+5)*sizeof(int));
        for(int i=1;i<=sz;i++)cnt[len[i]]++;
        for(int i=1;i<=sz;i++)cnt[i]+=cnt[i-1];
        for(int i=sz;i>=1;i--)ord[cnt[len[i]]--]=i;
    }
}G;
```

Splay

```
namespace Splay
{
    int sz[MAXN],ch[MAXN][2],fa[MAXN],val[MAXN],cnt[MAXN],rt,tot;
    void maintain(int u){sz[u]=sz[ch[u][0]]+sz[ch[u][1]]+cnt[u];}
    bool chside(int u){return u==ch[fa[u]][1];}
    void clear(int u){ch[u][0]=ch[u][1]=fa[u]=val[u]=sz[u]=cnt[u]=0;}
    void rotate(int u)
    {
        int f=fa[u],ff=fa[f],chk=chside(u);
        ch[f][chk]=ch[u][chk^1];
        fa[ch[u][chk^1]]=f;
        ch[u][chk^1]=f;
        fa[f]=u;
        fa[u]=ff;
        if(ff)ch[ff][f==ch[ff][1]]=u;
        maintain(f);
        maintain(u);
    }
    void splay(int u)
    {
        for(int f=fa[u];f=fa[u],f;rotate(u))
            if(fa[f])rotate(chside(u)==chside(f)?f:u);
        rt=u;
    }
    void ins(int k)
    {
        if(!rt)
        {
            val[++tot]=k;
            cnt[tot]++;
            rt=tot;
            maintain(rt);
            return;
        }
        int cur=rt,f=0;
        while(1)
        {
            if(val[cur]==k)
            {
                cnt[cur]++;
                maintain(cur);
                maintain(f);
                splay(cur);
                break;
            }
            f=cur;
            cur=ch[cur][val[cur]<k];
            if(!cur)
            {
                val[++tot]=k;
                cnt[tot]++;
                fa[tot]=f;
                ch[f][val[f]<k]=tot;
                maintain(tot);
                maintain(f);
                splay(tot);
                break;
            }
        }
    }
    int rk(int k)
    {
        int res=0,cur=rt;
        while(1)
        {
            if(k<val[cur])cur=ch[cur][0];
            else
            {
                res+=sz[ch[cur][0]];
                if(k==val[cur])
                {
                    splay(cur);
                    return res+1;
                }
                res+=cnt[cur];
                cur=ch[cur][1];
            }
        }
    }
    int kth(int k)
    {
        int cur=rt;
        while(1)
        {
            if(ch[cur][0]&&k<=sz[ch[cur][0]])cur=ch[cur][0];
            else
            {
                k-=cnt[cur];
                k-=sz[ch[cur][0]];
                if(k<=0)
                {
                    splay(cur);
                    return val[cur];
                }
                cur=ch[cur][1];
            }
        }
    }
    int pre()                // splay it to the root before using
    {
        int cur=ch[rt][0];
        while(ch[cur][1])cur=ch[cur][1];
        splay(cur);
        return cur;
    }
    int nxt()
    {
        int cur=ch[rt][1];
        while(ch[cur][0])cur=ch[cur][0];
        splay(cur);
        return cur;
    }
    void del(int k)
    {
        rk(k);
        if(cnt[rt]>1)
        {
            cnt[rt]--;
            maintain(rt);
            return;
        }
        if(!ch[rt][0]&&!ch[rt][1])
        {
            clear(rt);
            rt=0;
            return;
        }
        if(!(ch[rt][0]&&ch[rt][1]))
        {
            int cur=rt;
            if(ch[rt][1])rt=ch[rt][1];
            else rt=ch[rt][0];
            fa[rt]=0;
            clear(cur);
            return;
        }
        int cur=rt,u=pre();
        splay(u);
        fa[ch[cur][1]]=u;
        ch[u][1]=ch[cur][1];
        clear(cur);
        maintain(rt);
    }
}
```

Tarjan

```
struct tarjan_one_direction
{
    int head[MAXN],_next[MAXM],to[MAXM],ecnt;
    void add_Edge(int u,int v)
    {
        _next[++ecnt]=head[u];
        to[ecnt]=v;
        head[u]=ecnt;
    }
    int tot,dfn[MAXN],low[MAXN];
    int cir[MAXN],cirnum;
    stack<int> s;
    void tarjan(int u)
    {
        dfn[u]=low[u]=++tot;
        s.push(u);
        for(int i=head[u];i;i=_next[i])
        {
            int v=to[i];
            if(!dfn[v])
            {
                tarjan(v);
                low[u]=min(low[u],low[v]);
            }
            else if(!cir[v])low[u]=min(low[u],dfn[v]);
        }
        if(dfn[u]==low[u])
        {
            cir[u]=++cirnum;
            while(s.top()!=u)
            {
                cir[s.top()]=cirnum;
                s.pop();
            }
            s.pop();
        }
    }
    void solve()
    {
        for(int i=1;i<=n;i++)
            if(!dfn[i])tarjan(i);
    }
};
```

多项式 ln

```
#include<bits/stdc++.h>
using namespace std;
#define fi first
#define se second
#define SZ(x) ((int)x.size())
#define ALL(x) (x.begin(),x.end())
typedef long long ll;
typedef pair<int,int> pii;
typedef double db;
const int MAXN=6e5+10,MAXM=4e6+10;
const int INF=INT_MAX,SINF=0x3f3f3f3f;
const ll llINF=LLONG_MAX;
const int MOD=998244353,mod=998244353;
const int Lim=1<<20;
int add(int a,int b)
{
    return a+b>=MOD?a+b-MOD:a+b;
}

int sub (int a,int b)
{
    return a<b?a-b+MOD:a-b;
}

int mul (int a,int b)
{
    return (ll)a*b%MOD;
}

int ksm(int base,int v)
{
    int res=1;
    while(v)
    {
        if(v&1)res=mul(res,base);
        base=mul(base,base);
        v>>=1;
    }
    return res;
}

struct poly
{
    int a[Lim+5];
    int cnt;
}F,G,tmpinv;

int rev[Lim+5],lg2[Lim+5],rt[Lim+5],irt[Lim+5];
int n,k,lim,type;

void init()          //求原根
{
    for(int i=2;i<=Lim;i++)lg2[i]=lg2[i>>1]+1;
    rt[0]=1;
    rt[1]=ksm(3,(MOD-1)/Lim);    //第一个原根
    irt[0]=1;
    irt[1]=ksm(rt[1],MOD-2);   //费马小
    for(int i=2;i<=Lim;i++)
    {
        rt[i]=mul(rt[i-1],rt[1]);
        irt[i]=mul(irt[i-1],irt[1]);
    }
}

void NTT(poly &f,int type,int lim)
{
    int w,y,l=lg2[lim]-1;
    for(int i=1;i<lim;i++)
    {
        rev[i]=(rev[i>>1]>>1)|((i&1)<<l);
        if(i>=rev[i])continue;
        swap(f.a[i],f.a[rev[i]]);           //蝴蝶
    }
    l=Lim>>1;
    for(int mid=1;mid<lim;mid<<=1)
    {
        for(int j=0;j<lim;j+=(mid<<1))
        {
            for(int k=0;k<mid;k++)
            {
                w=type==1?rt[l*k]:irt[l*k];
                y=mul(w,f.a[j|k|mid]);
                f.a[j|k|mid]=sub(f.a[j|k],y);
                f.a[j|k]=add(f.a[j|k],y);
            }
        }
        l>>=1;
    }
    if(type==1)return;
    y=ksm(lim,MOD-2);
    for(int i=0;i<lim;i++)
        f.a[i]=mul(f.a[i],y);
}

void get_poly_inv(poly &f,int n,poly &g)
{
    fill(g.a,g.a+n+n,0);
    g.a[0]=ksm(f.a[0],MOD-2);
    for(int t=2;t<=n*2;t<<=1)
    {
        int t2=t<<1;
        copy(f.a,f.a+t,tmpinv.a);
        fill(tmpinv.a+t,tmpinv.a+t2,0);
        NTT(g,1,t2),NTT(tmpinv,1,t2);
        for(int i=0;i<t2;i++)g.a[i]=mul(g.a[i],sub(2,mul(g.a[i],tmpinv.a[i])));
        NTT(g,-1,t2);
        fill(g.a+t,g.a+t2,0);
    }
}

void get_Int(poly &f,int lim)  //积分
{
    for(int i=lim;i>0;i--)f.a[i]=mul(ksm(i,MOD-2),f.a[i-1]);f.a[0]=0;
}

void get_differentia(poly &f,int lim)  //微分
{
    for(int i=0;i<lim;i++)f.a[i]=mul(f.a[i+1],i+1);
}

void get_ln(poly &f,poly &g,int n)
{
    get_poly_inv(f,n,g);
    get_differentia(f,n);
    NTT(f,1,lim);NTT(g,1,lim);
    for(int i=0;i<lim;i++)G.a[i]=mul(g.a[i],f.a[i]);
    NTT(g,-1,lim);
    get_Int(g,n);
}

int main()
{
    init();
    int n;
    scanf("%d",&n);
    for(int i=0;i<n;i++)scanf("%d",&F.a[i]);
    lim=1;while(lim<=(n<<1))lim<<=1;
    get_ln(F,G,n);
    for(int i=0;i<n;i++)printf("%d ",G.a[i]);
    return 0;
}
```

多项式求逆

```
#include<bits/stdc++.h>
using namespace std;
#define fi first
#define se second
#define SZ(x) ((int)x.size())
#define ALL(x) (x.begin(),x.end())
typedef long long ll;
typedef pair<int,int> pii;
typedef double db;
const int MAXN=6e5+10,MAXM=4e6+10;
const int INF=INT_MAX,SINF=0x3f3f3f3f;
const ll llINF=LLONG_MAX;
const int MOD=998244353,mod=998244353;
const int Lim=1<<20;
int add(int a,int b)
{
    return a+b>=MOD?a+b-MOD:a+b;
}

int sub (int a,int b)
{
    return a<b?a-b+MOD:a-b;
}

int mul (int a,int b)
{
    return (ll)a*b%MOD;
}

int ksm(int base,int v)
{
    int res=1;
    while(v)
    {
        if(v&1)res=mul(res,base);
        base=mul(base,base);
        v>>=1;
    }
    return res;
}

struct poly
{
    int a[Lim+5];
    int cnt;
```

```
}F,G,tmpinv;

int rev[Lim+5],lg2[Lim+5],rt[Lim+5],irt[Lim+5];
int n,k,lim,type;

void init()         //求原根
{
    for(int i=2;i<=Lim;i++)lg2[i]=lg2[i>>1]+1;
    rt[0]=1;
    rt[1]=ksm(3,(MOD-1)/Lim);   //第一个原根
    irt[0]=1;
    irt[1]=ksm(rt[1],MOD-2);   //费马小
    for(int i=2;i<=Lim;i++)
    {
        rt[i]=mul(rt[i-1],rt[1]);
        irt[i]=mul(irt[i-1],irt[1]);
    }
}

void NTT(poly &f,int type,int lim)
{
    int w,y,l=lg2[lim]-1;
    for(int i=1;i<lim;i++)
    {
        rev[i]=(rev[i>>1]>>1)|((i&1)<<l);
        if(i>=rev[i])continue;
        swap(f.a[i],f.a[rev[i]]);           //蝴蝶
    }
    l=Lim>>1;
    for(int mid=1;mid<lim;mid<<=1)
    {
        for(int j=0;j<lim;j+=(mid<<1))
        {
            for(int k=0;k<mid;k++)
            {
                w=type==1?rt[l*k]:irt[l*k];
                y=mul(w,f.a[j|k|mid]);
                f.a[j|k|mid]=sub(f.a[j|k],y);
                f.a[j|k]=add(f.a[j|k],y);
            }
        }
        l>>=1;
    }
    if(type==1)return;
    y=ksm(lim,MOD-2);
    for(int i=0;i<lim;i++)
        f.a[i]=mul(f.a[i],y);
}

void get_poly_inv(poly &f,int n,poly &g)
{
    fill(g.a,g.a+n+n,0);
    g.a[0]=ksm(f.a[0],MOD-2);
    for(int t=2;t<=n*2;t<<=1)
    {
        int t2=t<<1;
        copy(f.a,f.a+t,tmpinv.a);
        fill(tmpinv.a+t,tmpinv.a+t2,0);
        NTT(g,1,t2),NTT(tmpinv,1,t2);
        for(int i=0;i<t2;i++)g.a[i]=mul(g.a[i],sub(2,mul(g.a[i],tmpinv.a[i])));
        NTT(g,-1,t2);
        fill(g.a+t,g.a+t2,0);
    }
}

int main()
{
    init();
    int n;
    scanf("%d",&n);
    for(int i=0;i<n;i++)scanf("%d",&F.a[i]);
    get_poly_inv(F,n,G);
    for(int i=0;i<n;i++)printf("%d ",G.a[i]);
    return 0;
}
```

分治 NTT

```
#include<bits/stdc++.h>
using namespace std;
#define fi first
#define se second
#define SZ(x) ((int)x.size())
#define ALL(x) (x.begin(),x.end())
typedef long long ll;
typedef pair<int,int> pii;
typedef double db;
const int MAXN=8e5+10,MAXM=4e6+10;
const int INF=INT_MAX,SINF=0x3f3f3f3f;
const ll llINF=LLONG_MAX;
const int MOD=998244353,mod=119;
const int Lim=1<<20;

int add(int a,int b)
{
    return a+b>=MOD?a+b-MOD:a+b;
}

int sub (int a,int b)
{
    return a<b?a-b+MOD:a-b;
}

int mul (int a,int b)
{
    return (ll)a*b%MOD;
}

int ksm(int base,int v)
{
    int res=1;
    while(v)
    {
        if(v&1)res=mul(res,base);
        base=mul(base,base);
        v>>=1;
    }
    return res;
}

struct poly
{
    int a[Lim+5];
    int cnt;
}F,G;

int rev[Lim+5],lg2[Lim+5],rt[Lim+5],irt[Lim+5];
int n,k,lim,type,a[MAXN],b[MAXN];

void init()         //求原根
{
    for(int i=2;i<=Lim;i++)lg2[i]=lg2[i>>1]+1;
    rt[0]=1;
    rt[1]=ksm(3,(MOD-1)/Lim);   //第一个原根
    irt[0]=1;
    irt[1]=ksm(rt[1],MOD-2);   //费马小
    for(int i=2;i<=Lim;i++)
    {
        rt[i]=mul(rt[i-1],rt[1]);
        irt[i]=mul(irt[i-1],irt[1]);
    }
}

void NTT(poly &f,int type,int lim)
{
    int w,y,l=lg2[lim]-1;
    for(int i=1;i<lim;i++)
    {
        rev[i]=(rev[i>>1]>>1)|((i&1)<<l);
        if(i>=rev[i])continue;
        swap(f.a[i],f.a[rev[i]]);           //蝴蝶
    }
    l=Lim>>1;
    for(int mid=1;mid<lim;mid<<=1)
    {
        for(int j=0;j<lim;j+=(mid<<1))
        {
            for(int k=0;k<mid;k++)
            {
                w=type==1?rt[l*k]:irt[l*k];
                y=mul(w,f.a[j|k|mid]);
                f.a[j|k|mid]=sub(f.a[j|k],y);
                f.a[j|k]=add(f.a[j|k],y);
            }
        }
        l>>=1;
    }
    if(type==1)return;
    y=ksm(lim,MOD-2);
    for(int i=0;i<lim;i++)
        f.a[i]=mul(f.a[i],y);
}
```

```
void solve(int l,int r,int lim)
{
    if(lim<=1)return;
    if(l>=n)return;
    int mid=(l+r)>>1;
    solve(l,mid,lim>>1);
    memset(F.a+lim/2,0,sizeof(int)*lim);
    memcpy(F.a,a+l,sizeof(int)*lim/2);
    memcpy(G.a,b,sizeof(int)*lim);
    NTT(F,1,lim),NTT(G,1,lim);
    for(int i=0;i<r-l;i++)F.a[i]=mul(F.a[i],G.a[i]);
    NTT(F,-1,lim);
    for(int i=(r-l)/2;i<r-l;i++)a[l+i]=add(a[l+i],F.a[i]);
    solve(mid,r,lim>>1);
}

int main()
{
    init();
    scanf("%d",&n);
    for(int i=1;i<n;i++)scanf("%d",&b[i]);
    a[0]=1;
    lim=1;
    while(lim<=(n<<1))lim<<=1;
    solve(0,lim,lim);
    for (int i=0;i<n;i++) printf("%d%c",a[i],i==n-1?'\n':' ');
    return 0;
}
```

割点

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef pair<int,int> pii;
const int MAXN=3000500,INF=~(1<<31),MOD=1e9+7,mod=998244353;
const ll llINF=~(1ll<<63);
#define lowbit(x) x&-x

int head[MAXN],to[MAXN],_next[MAXN],ecnt;

void add_edge(int u,int v)
{
    _next[++ecnt]=head[u];
    to[ecnt]=v;
    head[u]=ecnt;
}

int dfn[MAXN],low[MAXN],tot,cut[MAXN];

void tarjan(int u,int fa)
{
    dfn[u]=low[u]=++tot;
    int ch=0;
    for(int i=head[u];i;i=_next[i])
    {
        int v=to[i];
        if(!dfn[v])
        {
            tarjan(v,fa);
            low[u]=min(low[u],low[v]);
            if(low[v]>=dfn[u]&&u!=fa)cut[u]=1;
            if(u==fa)ch++;
        }
        low[u]=min(low[u],dfn[v]);
    }
    if(u==fa&&ch>=2)cut[fa]=1;
}

int main()
{
    int n,m,u,v;
    scanf("%d%d",&n,&m);
    for(int i=1;i<=m;i++)
    {
        scanf("%d%d",&u,&v);
        add_edge(u,v);
        add_edge(v,u);
    }
    for(int i=1;i<=n;i++)
        if(!dfn[i])tarjan(i,i);
    int ans=0;
    for(int i=1;i<=n;i++)ans+=cut[i];
    printf("%d\n",ans);
    for(int i=1;i<=n;i++)if(cut[i])printf("%d ",i);
    return 0;
}
```

回文自动机

```
struct PAM
{
    int sz,tot,last;
    int ch[MAXN][26],len[MAXN],fail[MAXN];
    int cnt[MAXN],dep[MAXN],dif[MAXN],slink[MAXN],baby[MAXN];
    char s[MAXN];
    int newnode(int l)
    {
        sz++;
        memset(ch[sz],0,sizeof(ch[sz]));
        len[sz]=l;
        fail[sz]=cnt[sz]=dep[sz]=0;
        return sz;
    }
    void init()
    {
        sz=-1;
        last=0;
        s[tot=0]='$';
        newnode(0);
        newnode(-1);
        fail[0]=1;
    }
    int getfail(int x)
    {
        while(s[tot-len[x]-1]!=s[tot])x=fail[x];
        return x;
    }
    void insert(char c)
    {
        s[++tot]=c;
        int now=getfail(last);
        if(!ch[now][c-'a'])
        {
            int x=newnode(len[now]+2);
            fail[x]=ch[getfail(fail[now])][c-'a'];
            dep[x]=dep[fail[x]]+1;
            ch[now][c-'a']=x;
            dif[x]=len[x]-len[fail[x]];

 if(dif[x]==dif[fail[x]])slink[x]=slink[fail[x]],baby[x]=baby[fail[x]];
            else slink[x]=fail[x],baby[x]=x;
        }
        last=ch[now][c-'a'];
        cnt[last]++;
    }
}pam;
```

静态区间第 k 大（主席树）

```
#include<bits/stdc++.h>
using namespace std;
#define fi first
#define se second
#define SZ(x) ((int)x.size())
#define ALL(x) (x.begin(),x.end())
typedef long long ll;
typedef pair<int,int> pii;
typedef double db;
const int MAXN=6e5+10,MAXM=4e6+10;
const int INF=INT_MAX,SINF=0x3f3f3f3f;
const ll llINF=LLONG_MAX;
const int MOD=1e9+7,mod=1e9+7;
const int Lim=1<<20;

int n,m,a[MAXN],len,ind[MAXN],tot;
int sum[MAXN<<4],rt[MAXN<<4],ls[MAXN<<4],rs[MAXN<<4];

int getid(int u)
{
    return lower_bound(ind+1,ind+len+1,u)-ind;
}

int init(int l,int r)
{
    int root=++tot;
    if(l==r)return root;
    int mid=(l+r)>>1;
    ls[tot]=init(l,mid);
    rs[tot]=init(mid+1,r);
    return root;
}

int update(int l,int r,int k,int root)
{
    int dir=++tot;
    ls[dir]=ls[root];
    rs[dir]=rs[root];
    sum[dir]=sum[root]+1;
    if(l==r)return dir;
    int mid=(l+r)>>1;
    if(k<=mid)ls[dir]=update(l,mid,k,ls[dir]);
    else      rs[dir]=update(mid+1,r,k,rs[dir]);
    return dir;
}

int query(int u,int v,int l,int r,int k)
{
    int mid=(l+r)>>1;
    int x=sum[ls[v]]-sum[ls[u]];
    if(l==r)return l;
    if(k<=x)return query(ls[u],ls[v],l,mid,k);
    else return query(rs[u],rs[v],mid+1,r,k-x);
}

int main()
{
    scanf("%d%d",&n,&m);
    for(int i=1;i<=n;i++)scanf("%d",&a[i]);
    memcpy(ind,a,sizeof(ind));
    sort(ind+1,ind+n+1);
    len=unique(ind+1,ind+n+1)-ind-1;
    rt[0]=init(1,len);
    for(int i=1;i<=n;i++)rt[i]=update(1,len,getid(a[i]),rt[i-1]);
    for(int i=1;i<=m;i++)
    {
        int l,r,k;
        scanf("%d%d%d",&l,&r,&k);
        printf("%d\n",ind[query(rt[l-1],rt[r],1,len,k)]);
    }
    return 0;
}
```

矩阵树定理（无向边）

```
struct Matrix
{
    ll a[505][505],n,p;
    void add(int u,int v)
    {
        a[u][v]--;a[v][u]--;
        a[u][u]++;a[v][v]++;
    }
    ll Guass()
    {
        for(int i=1;i<=n;i++)
          for(int j=1;j<=n;j++)  a[i][j]=(a[i][j]+p)%p;
        ll res=1;
        for(int i=1;i<n;i++)
        {
            int pos=i;
            while(pos<n&&!a[pos][i])  pos++;
            if(pos>=n)  continue;
            if(pos!=i)  res=p-res,swap(a[pos],a[i]);
            int inv=ksm(a[i][i],p-2);
            for(int j=i+1;j<n;j++)
            {
                ll tmp=1ll*inv*a[j][i]%p;
                for(int k=i;k<n;k++)  a[j][k]=(a[j][k]-1ll*a[i][k]*tmp%p+p)%p;
            }
        }
        for(int i=1;i<n;i++)  res=1ll*res*a[i][i]%p;
        return res;
    }
}G;
```

拉格朗日插值

```
int n,k,x[MAXN],y[MAXN],num[MAXN],tmp[MAXN],res[MAXN],inv[MAXN];

int ksm(int x,int v)
{
    int ans=1;
    for(;v;v>>=1,x=1ll*x*x%mod)if(v&1)ans=1ll*ans*x%mod;
    return ans;
}

int Inv(int x)
{
    return ksm(x,mod-2);
}

void pre()
{
    num[0]=1;
    for(int i=1;i<=n;i++,swap(num,tmp))
    {
        tmp[0]=0;
        inv[i]=Inv(mod-x[i]);
        for(int j=1;j<=i;j++)tmp[j]=num[j-1];
        for(int j=0;j<=i;j++)(tmp[j]=tmp[j]+1ll*num[j]*(mod-x[i])%mod)%=mod;
    }
}

void Lagrange()
{
    for(int i=1;i<=n;i++)
    {
        int den=1,lst=0;
        for(int j=1;j<=n;j++)if(i^j)den=1ll*den*(x[i]-x[j]+mod)%mod;
        den=1ll*y[i]*Inv(den)%mod;
        for(int j=0;j<n;j++)
        {
            tmp[j]=1ll*(num[j]-lst+mod)*inv[i]%mod;
            res[j]=(res[j]+1ll*den*tmp[j]%mod)%mod;
```

```
                lst=tmp[j];
            }
        }
}

int calc(int x)
{
    ll ret=0,var=1;
    for(int i=0;i<n;var=var*x%mod,i++)ret=(ret+var*res[i])%mod;
    return ret;
}

int main()
{
    scanf("%d%d",&n,&k);
    for(int i=1;i<=n;i++)scanf("%d%d",&x[i],&y[i]);
    pre();
    Lagrange();
    printf("%d\n",calc(k));
    return 0;
}
```

拉格朗日插值 $O(n)$ 版（要求 x 取值连续）

```
ll lagrange(ll n, ll xx)
{
    xx %= M;
    inv[0] = inv[1] = 1;
    for (int i = 2; i <= n * 2; i++)
        inv[i] = (M - M / i) * inv[M % i] % M;
    x[0] = 0, y[0] = 0;
    for (int i = 1; i <= n; i++)
    {
        x[i] = i, y[i] = f(i) % M;
        if (xx == i)
            return y[i];
    }
    ll ret = 0;
    suf[n + 1] = 1, pre[0] = (xx - x[0] + M) % M;
    for (int i = 1; i <= n; i++)
        pre[i] = pre[i - 1] * (xx - x[i]) % M;
    for (int i = n; ~i; i--)
        suf[i] = suf[i + 1] * (xx - x[i]) % M;
    for (int i = 2; i <= n; i++)
        inv[i] = inv[i] * inv[i - 1] % M;
    for (int i = 0; i <= n; i++)
        (ret += y[i] * (i == 0 ? 1 : pre[i - 1]) % M * suf[i + 1] % M * inv[i] %
M * (((n - i) & 1) ? M - 1 : 1) % M * inv[n - i] % M) %= M;
    return ret;
}
```

莫队

```
struct MOalgorithm
{
    int cnt;
    struct QUERY
    {
        int l,r,id;
        bool operator < (const QUERY &rhs)const
        {
            return block[l]==block[rhs.l]?block[l]&1?r>rhs.r:r<rhs.r:l<rhs.l;
        }
    }q[MAXN];
    void add(int x)
    {

    }
    void del(int x)
    {

    }
    void solve()
    {
        int l=1,r=0;
        sort(q+1,q+cnt+1);
        for(int i=1;i<=cnt;i++)
        {
            while(l<q[i].l)del(l++);
            while(r>q[i].r)del(r--);
            while(r<q[i].r)add(++r);
            while(l>q[i].l)add(--l);
            ans[q[i].id]=tot;
        }
    }
};
```

三维偏序（CDQ）

```
struct Node
{
    int x,y,z;
    bool b;
    int *ans;
    void gi()
    {
        scanf("%d%d%d",&x,&y,&z);
    }
    bool operator == (const Node &a)const
    {
        return x==a.x&&y==a.y&&z==a.z;
    }
}a[MAXN],b[MAXN],c[MAXN];

bool cmp(Node &a,Node &b)
{
    if(b.x^a.x)return a.x<b.x;
    if(b.y^a.y)return a.y<b.y;
    return a.z<b.z;
}

int n,k,ans[MAXN]={0},d[MAXN]={0};bool f=0;
```

```
void merge2(int l,int r)
{
    if(l==r)return;
    int mid=(l+r)>>1;
    merge2(l,mid);
    merge2(mid+1,r);
    for(int i=l,j=l,k=mid+1,cnt=0;i<=r;i++)
    {
        if((k>r||b[j].z<=b[k].z)&&j<=mid)c[i]=b[j++],cnt+=c[i].b;
        else
        {
            c[i]=b[k++];
            if(!c[i].b)*c[i].ans+=cnt;
        }
    }
    for(int i=l;i<=r;i++)b[i]=c[i];
}

void merge1(int l,int r)
{
    if(l==r)return;
    int mid=(l+r)>>1;
    merge1(l,mid);
    merge1(mid+1,r);
    for(int i=l,j=l,k=mid+1;i<=r;i++)
    {
        if((k>r||a[j].y<=a[k].y)&&j<=mid)b[i]=a[j++],b[i].b=1;
        else b[i]=a[k++],b[i].b=0;
    }
    for(int i=l;i<=r;i++)a[i]=b[i];
    if(l==1&&r==5)f=1;
    merge2(l,r);
}

int main()
{
    scanf("%d%d",&n,&k);
    for(int i=1;i<=n;i++)
    {
        a[i].gi();
        a[i].ans=&ans[i];
        ans[i]=0;
    }
    sort(a+1,a+n+1,cmp);
    for(int i=n-1;i;i--)if(a[i]==a[i+1])
        *a[i].ans=*a[i+1].ans+1;
    merge1(1,n);
    for(int i=1;i<=n;i++)d[ans[i]]++;
    for(int i=0;i<n;i++)printf("%d\n",d[i]);
    return 0;
}
```

线性基

```
struct Linear_base
{
    ll num[70];
    bool f=0;
    bool insert(ll x)
    {
        for(ll i=63;i>=0;i--)
        if((x>>i)&1)
        {
            if(!num[i])
            {
                num[i]=x;
                return 1;
            }
            x^=num[i];
        }
        f=1;
        return 0;
    }
    ll qmax(ll x)
    {
        ll res=x;
        for(ll i=63;i>=0;i--)
        {
            if(res^num[i]>res)
                res^=num[i];
        }
        return res;
    }
    ll qmin()
    {
        if(f)return 0;
        for(int i=0;i<=63;i++)
            if(num[i])return num[i];
    }
    ll queryk(ll k)
    {
        ll res=0,cnt=0;
        vector<ll> tmp;
        k-=f;
        if(!k)return 0;
        for(ll i=0;i<=63;i++)
        {
            for(int j=i-1;~j;j--)
                if(num[i]&(1ll<<j))num[i]^=num[j];
            if(num[i])tmp.push_back(num[i]);
        }
        if(k>=(1ll<<SZ(tmp)))return -1;
        for(auto j:tmp)if(k&(1ll<<(cnt++)))res^=j;
        return res;
    }
}L;
```

虚树

```
#include<bits/stdc++.h>
using namespace std;
struct Edge
{
    int v,nxt;
```

```
}e[MAXM];

int head[MAXN],ecnt;

void add_edge(int u,int v)
{
    e[++ecnt]={v,head[u]};head[u]=ecnt;
}

int dep[MAXN],fa[MAXN][20],dfn[MAXN],dfncnt,n,m;

void LCA_pre_dfs(int u,int f,int d)
{
    dep[u]=d;fa[u][0]=f;dfn[u]=++dfncnt;
    for(int i=head[u];i;i=e[i].nxt)
    {
        if(e[i].v!=f)LCA_pre_dfs(e[i].v,u,d+1);
    }
}

void LCA_pre()
{
    LCA_pre_dfs(1,0,1);
    for(int j=1;j<=19;j++)
    for(int i=1;i<=n;i++)fa[i][j]=fa[fa[i][j-1]][j-1];
}

int LCA(int u,int v)
{
    if(dep[u]<dep[v])swap(u,v);
    int delta=dep[u]-dep[v];
    for(int i=19;i>=0;i--)
    if((delta>>i)&1)u=fa[u][i];
    for(int i=19;i>=0;i--)
        if(fa[u][i]^fa[v][i])
    {
        u=fa[u][i];
        v=fa[v][i];
    }
    if(u==v)return u;
    return fa[u][0];
}

int stk[MAXN],top;
int que[MAXN],vis[MAXN];

void get_tree(int n)
{
    stk[top=1]=1;head[1]=0; // decide a root
    for(int i=1;i<=n;i++)
    {
        if(que[i]==1)continue;
        int fa=LCA(stk[top],que[i]);
        if(fa!=stk[top])
        {
            while(dfn[stk[top-1]]>dfn[fa])
            {
                add_edge(stk[top-1],stk[top]);
                top--;
            }
            if(dfn[stk[top-1]]!=dfn[fa])
            {
                head[fa]=0;
                add_edge(fa,stk[top]);
                stk[top]=fa;
            }
            else add_edge(fa,stk[top--]);
        }
        head[que[i]]=0;
        stk[++top]=que[i];
    }
    for(int i=1;i<top;i++)add_edge(stk[i],stk[i+1]);
}

// Guide: Be cautious about n in LCA_pre

void solve()
{
    int k;
    scanf("%d",&k);
    for(int i=1;i<=k;i++)scanf("%d",&que[i]),vis[que[i]]=1; // key points
    sort(que+1,que+k+1,[&](int a,int b){return dfn[a]<dfn[b];});
    get_tree(k);
    tree_DP(1,0);   // root is 1
    for(int i=1;i<=k;i++)vis[que[i]]=0;
}

int main()
{
    scanf("%d",&n);
    for(int i=1;i<n;i++)
    {
        int u,v;scanf("%d%d",&u,&v);
        add_edge(u,v);
        add_edge(v,u);
    }
    LCA_pre();
    scanf("%d",&m);
    while(m--)solve();
    return 0;
}
```

悬线法

```
for(int i=1;i<=n;i++)
    for(int j=1;j<=m;j++)
    {
        l[i][j]=r[i][j]=j,u[i][j]=1;
    }
for(int i=1;i<=n;i++)
{
    for(int j=1;j<=m;j++)
    if(ok(i,j)&&ok(i,j-1))l[i][j]=l[i][j-1];
    for(int j=m;j;j--)
    if(ok(i,j)&&ok(i,j+1))r[i][j]=r[i][j+1];
}
for(int i=1;i<=n;i++)
    for(int j=1;j<=m;j++)
    {
        if(i>1&&ok(i,j)&&ok(i-1,j))
        {
            l[i][j]=max(l[i][j],l[i-1][j]);
            r[i][j]=min(r[i][j],r[i-1][j]);
            u[i][j]=u[i-1][j]+1;
        }
        if(check(i,j)&&calc(i,j)>calc(x,y))x=i,y=j;
    }
```

原根

```
int ksm(int x,int v,int mod)
{
    int res=1;
    while(v)
    {
        if(v&1)res=1ll*res*x%mod;
        x=1ll*x*x%mod;
        v>>=1;
    }
    return res;
}

int getRoot(int x)
{
    vector<int> fac;
    int tot=0;
    int phi=x-1;
    for(int i=2;i*i<=phi;i++)
        if(phi%i==0)
        {
            fac.push_back(i);
            while(phi%i==0)phi/=i;
        }
    if(phi>1)fac.push_back(phi);
    phi=x-1;
    for(int i=2;i<=phi;i++)
    {
        bool f=1;
        for(auto j:fac)
        {
            if(ksm(i,phi/j,x)==1)f=0;
            if(!f)break;
        }
        if(f)return i;
    }
    return -1;
}
```

子序列自动机 $O(\log n)$

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

```
using namespace std;
#define fi first
#define se second
#define SZ(x) ((int)x.size())
#define ALL(x) (x.begin(),x.end())
typedef long long ll;
typedef pair<int,int> pii;
typedef double db;
const int MAXN=6e5+10,MAXM=4e6+10;
const int INF=INT_MAX,SINF=0x3f3f3f3f;
const ll llINF=LLONG_MAX;
const int MOD=1e9+7,mod=998244353;

vector<int> v[MAXN];
int a[MAXN],b[MAXN];
int n,m,q,type;

bool check(int l)
{
    int u=0;
    for(int i=1;i<=l;i++)
    {
        int tu=upper_bound(v[b[i]].begin(),v[b[i]].end(),u)-v[b[i]].begin();
        if(tu==SZ(v[b[i]]))return false;
        u=v[b[i]][tu];
    }
    return true;
}

int main()
{
    scanf("%d%d%d%d",&type,&n,&q,&m);
    for(int i=1;i<=n;i++)scanf("%d",&a[i]);
    for(int i=1;i<=n;i++)v[a[i]].push_back(i);
    for(int i=1;i<=q;i++)
    {
        int l;
        scanf("%d",&l);
        for(int j=1;j<=l;j++)scanf("%d",&b[j]);
        if(check(l))puts("Yes");
        else puts("No");
    }
    return 0;
}
```

HLPP

```
int h[MAXN];
struct hlpp
{
    int n,s,t;
    int ecnt,inq[MAXN],gap[MAXN<<1],head[MAXN],flow[MAXN];
    struct Edge
    {
        int nxt,v,f;
    }e[MAXM];
    struct cmp
    {
        inline bool operator ()(int a,int b)
        {
            return h[a]<h[b];
        }
    };
    queue<int> q;
    priority_queue<int,vector<int>,cmp> pq;
    void init()
    {
        memset(head,-1,sizeof(head));
        memset(gap,0,sizeof(gap));
        ecnt=-1;
    }
    inline void add_Edge(int u,int v,int f)
    {
        e[++ecnt]={head[u],v,f};head[u]=ecnt;
        e[++ecnt]={head[v],u,0};head[v]=ecnt;
    }
    bool bfs()
    {
        memset(h,0x3f,(n+5)*sizeof(int));
        h[t]=0;
        q.push(t);
        while(!q.empty())
        {
            int u=q.front();q.pop();
            for(Rint i=head[u];~i;i=e[i].nxt)
                if(e[i^1].f&&h[e[i].v]>h[u]+1)
                h[e[i].v]=h[u]+1,q.push(e[i].v);
        }
        return h[s]!=SINF;
    }
    void push(int u)
    {
        for(Rint i=head[u];~i;i=e[i].nxt)
            if(e[i].f&&h[e[i].v]+1==h[u])
            {
                int d=std::min(flow[u],e[i].f),v=e[i].v;
                e[i].f-=d;
                e[i^1].f+=d;
                flow[u]-=d;
                flow[v]+=d;
                if(v!=s&&v!=t&&!inq[v])
                    pq.push(v),inq[v]=1;
                if(!flow[u])break;
            }
    }
    void relabel(int u)
    {
        h[u]=SINF;
        for(Rint i=head[u];~i;i=e[i].nxt)
            if(e[i].f&&h[e[i].v]+1<h[u])
            h[u]=h[e[i].v]+1;
    }
    int solve()
    {
        if(!bfs())return 0;
        h[s]=n;
        for(Rint i=1;i<=n;i++)if(h[i]<SINF)gap[h[i]]++;
        for(Rint i=head[s];~i;i=e[i].nxt)if(h[e[i].v]<SINF)
        {
            int d=e[i].f;
            if(!d)continue;
            e[i].f-=d;
            e[i^1].f+=d;
            flow[s]-=d;
            flow[e[i].v]+=d;
            if(e[i].v!=s&&e[i].v!=t&&!inq[e[i].v])pq.push(e[i].v),inq[e[i].v]=1;
        }
        while(!pq.empty())
        {
            int u=pq.top();pq.pop();
            inq[u]=0;push(u);
            if(flow[u])
            {
                if(!--gap[h[u]])
                {
                    for(Rint i=1;i<=n;i++)
                        if(i!=s&&i!=t&&h[i]>h[u]&&h[i]<n+1)h[i]=n+1;
                }
                relabel(u);++gap[h[u]];
                pq.push(u);inq[u]=1;
            }
        }
        return flow[t];
    }
}G;
```

ISAP

```
struct ISAP {
    struct Edge {
        ll v, w, nxt;
    } edge[MAXM];
    ll tot, num, s, t,n;
    ll head[MAXN];
    void init() {
        memset(head, -1, sizeof(head));
        tot = 0;
    }
    void add_Edge(ll u, ll v, ll w) {
        edge[tot].v = v;
        edge[tot].w = w;
        edge[tot].nxt = head[u];
        head[u] = tot++;
        edge[tot].v = u;
        edge[tot].w = 0;
        edge[tot].nxt = head[v];
        head[v] = tot++;
    }

    ll  d[MAXN], vis[MAXN], gap[MAXN];
    void bfs() {
        memset(d, 0, (n+5)*sizeof(ll));
        memset(gap, 0, (n+5)*sizeof(ll));
        memset(vis, 0, (n+5)*sizeof(ll));
        queue<int>q;
        q.push(t);
        vis[t] = 1;
        while (!q.empty()) {
            int u = q.front();
            q.pop();
            for (int i = head[u]; ~i; i = edge[i].nxt) {
                int v = edge[i].v;
                if (!vis[v]) {
                    d[v] = d[u] + 1;
                    gap[d[v]]++;
                    q.push(v);
                    vis[v] = 1;
                }
            }
        }
    }

    ll last[MAXN];
    ll dfs(int u, ll f) {
        if (u == t) return f;
        ll sap = 0;
        for (int i = last[u]; ~i; i = edge[i].nxt) {
            int v = edge[i].v;
            if (edge[i].w > 0 && d[u] == d[v] + 1) {
                last[u] = i;
                ll tmp = dfs(v, min(f - sap, edge[i].w));
                edge[i].w -= tmp;
                edge[i ^ 1].w += tmp;
                sap += tmp;
                if (sap == f) return sap;
            }
        }
        if (d[s] >= num) return sap;
        if (!(--gap[d[u]])) d[s] = num;
        ++gap[++d[u]];
        last[u] = head[u];
        return sap;
    }

    ll solve(int st, int ed, int n) {
        ll flow = 0;
        num = n;
        s = st;
        t = ed;
        bfs();
        memcpy(last, head, sizeof(head));
        while (d[s] < num) flow += dfs(s, INF);
        return flow;
    }
} G;
```