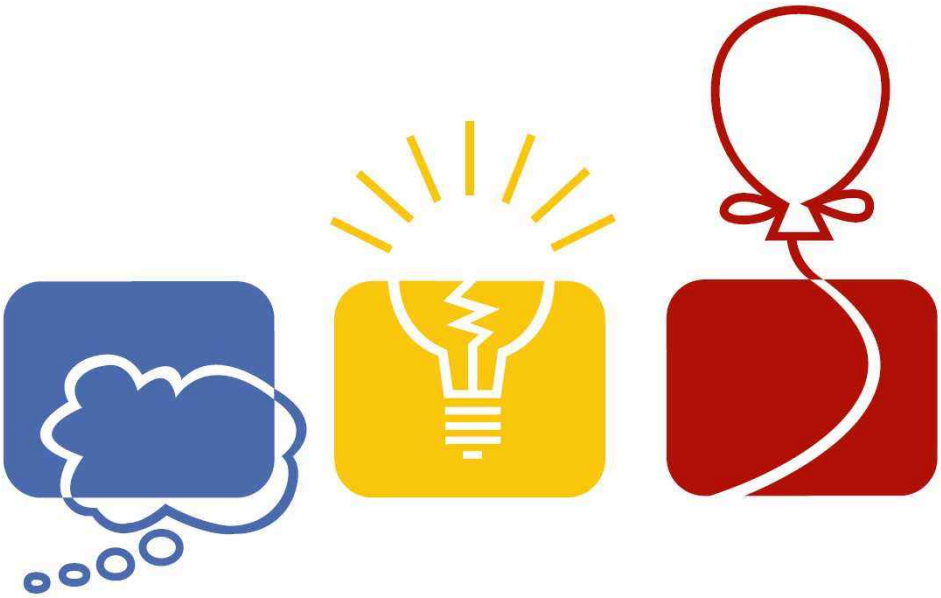

ACM/ICPC Template Manaual

QUST

QY

October 31, 2018

Contents

0 Math

0.1 Pow

```
long long pow(long long a,int k,long long mod)
{
    long long b=1;
    while(k){if(k&1) b=b*a%mod;a=a*a%mod;k>>=1;}
    return b;
}
```

0.2 Matrix

```
struct Mat{long long m[3][3];Mat(){memset(m,0,sizeof(m));}};
Mat multi(const Mat &a,const Mat &b)
{
    Mat c;
    for(int i=0;i<3;i++)
    for(int j=0;j<3;j++)if(a.m[j][i]!=0)
    for(int k=0;k<3;k++)if(b.m[i][k]!=0)
    c.m[j][k]=(c.m[j][k]+a.m[j][i]*b.m[i][k]%MOD)%MOD;
    return c;
}
Mat pow(Mat &a,int k)
{
    Mat b;
    for(int i=0;i<3;i++) b.m[i][i]=1;
    while(k){if(k&1) b=multi(b,a);a=multi(a,a);k>>=1;}
    return b;
}
```

0.3 Gcd_Lcm

```
long long gcd(long long x,long long y){return y==0?x:gcd(y,x%y);}
long long lcm(long long x,long long y){return x/gcd(x,y)*y;}
```

0.4 FFT

```
const int MAX=1<<17;
const double PI=acos(-1.0);
typedef complex<double> C;
map<int,int>mp;map<int,int>::iterator p;
C a1[MAX],a2[MAX],a3[MAX],ans[MAX];int n,L,cnt[MAX];
void rader(C *c_)
{
    for(int i=1,j=L/2,k;i<L-1;++i)
    {
        if(i<j) swap(c_[i],c_[j]);
        k=L/2;
        while(j>=k){j-=k;k/=2;}
        if(j<k) j+=k;
    }
}
void fft(C *c_,int v)
{
    rader(c_);
```

```
    for(int i=2;i<=L;i<<=1)
    {
        C wn(cos(-2.0*v*PI/i),sin(-2.0*v*PI/i));
        for(int j=0;j<L;j+=i)
        {
            C w(1,0);
            for(int k=j;k<j+i/2;++k)
            {
                C u=c_[k];
                C t=w*c_[k+i/2];
                c_[k]=u+t;
                c_[k+i/2]=u-t;
                w=w*wn;
            }
        }
    }
    if(v==-1) for(int i=0;i<L;++i) c_[i].real(c_[i].real()/L);
}
int main()
{
    int x;scanf("%d",&n);mp.clear();
    for(int i=0;i<n;++i)
    {
        scanf("%d",&x);x+=20000;
        if(mp[x]) ++mp[x];
        else mp[x]=1;
    }
    for(p=mp.begin();p!=mp.end();++p) a1[p->first]=a2[p->first*2]=a3[p->first*3]=C(p->
    second,0);L=MAX;
    /*L=1;while(L<L1<<1||L<L2<<1) L<<=1;
      L=L1+L2-1;while(ans[L]<=0&&L>0) --L;*/
    fft(a1,1);fft(a2,1);fft(a3,1);
    for(int i=0;i<L;++i) ans[i]=a1[i]*a1[i]*a1[i]-a1[i]*a2[i]*3.0+a3[i]*2.0;
    fft(ans,-1);
    for(int i=0;i<L;++i)
    {
        long long tmp=(long long)(ans[i].real()+0.5)/6;
        if(tmp) printf("%d : %I64d\n",i-60000,tmp);
    }
    return 0;
}
```

0.5 NTT

```
/* MOD:469762049□998244353□1004535809 ;this problem need pow*/
const int g=3;//in this problem
long long fact[MAX],inv[MAX],bit[MAX],a[MAX<<1],b[MAX<<1],c[MAX],ni;int t,n,L;
void init()
{
    fact[0]=inv[0]=bit[0]=1;
    for(int i=1;i<MAX;i++)
    {
        fact[i]=fact[i-1]*i%MOD;
        inv[i]=inv[i-1]*pow(i,MOD-2,MOD)%MOD;
        bit[i]=bit[i-1]*2%MOD;
    }
}
void rader(long long *f_)
```

```
{
    int i,j,k;
    for(i=1,j=L/2;i<L-1;++i)
    {
        if(i<j) swap(f_[i],f_[j]);
        k=L/2;
        while(j>=k){j-=k;k>>=1;}
        if(j<k) j+=k;
    }
}
void ntt(long long *f_,int t)
{
    rader(f_);
    for(int i=2;i<=L;i<<=1)
    {
        long long wn=pow(g,(MOD-1)/i,MOD);
        if(t==-1) wn=pow(wn,MOD-2,MOD);
        for(int j=0;j<L;j+=i)
        {
            long long e=1;
            for(int k=j;k<j+i/2;++k)
            {
                long long u=f_[k];
                long long v=e*f_[k+i/2]%MOD;
                f_[k]=(u+v)%MOD;
                f_[k+i/2]=(u-v+MOD)%MOD;
                e=e*wn%MOD;
            }
        }
    }
    if(t==-1) for(int i=0;i<L;++i) f_[i]=f_[i]*ni%MOD;
}
int main()
{
    init();scanf("%d",&t);
    while(t--)
    {
        memset(a,0,sizeof(a));memset(b,0,sizeof(b));
        scanf("%d",&n);L=1;
        while(L<n<<1) L<<=1;
        ni=pow(L,MOD-2,MOD);
        for(int i=1;i<=n;++i) scanf("%I64d",&c[i]);
        sort(c+1,c+n+1,greater<long long>());
        for(int i=0;i<n;++i)
        {
            a[i]=bit[n-i]*inv[i]%MOD;
            b[i]=c[n-i]*fact[n-i-1]%MOD;
        }
        ntt(a,1);ntt(b,1);
        for(int i=0;i<L;++i) a[i]=a[i]*b[i]%MOD;
        ntt(a,-1);
        long long r=inv[2],ans=0;
        for(int i=1;i<=n;++i)
        {
            ans=(ans+a[n-i]*inv[i-1]%MOD*r%MOD)%MOD;
            r=r*inv[2]%MOD;
            printf("%I64d ",ans);
        }
    }
```

```
    return 0;
}
```

0.6 FWT

```
/* opt=1/-1,n=1<<? */
void FWT_or(int *a,int opt)
{
    for(int i=1;i<N;i<<=1)
    for(int p=i<<1,j=0;j<N;j+=p)
    for(int k=0;k<i;++k)
        if(opt==1)a[i+j+k]=(a[j+k]+a[i+j+k])%MOD;
        else a[i+j+k]=(a[i+j+k]+MOD-a[j+k])%MOD;
}
void FWT_and(int *a,int opt)
{
    for(int i=1;i<N;i<<=1)
    for(int p=i<<1,j=0;j<N;j+=p)
    for(int k=0;k<i;++k)
        if(opt==1)a[j+k]=(a[j+k]+a[i+j+k])%MOD;
        else a[j+k]=(a[j+k]+MOD-a[i+j+k])%MOD;
}
void FWT_xor(int *a,int opt)
{
    for(int i=1;i<N;i<<=1)
    for(int p=i<<1,j=0;j<N;j+=p)
    for(int k=0;k<i;++k)
    {
        int X=a[j+k],Y=a[i+j+k];
        a[j+k]=(X+Y)%MOD;a[i+j+k]=(X+MOD-Y)%MOD;
        if(opt==-1)a[j+k]=1ll*a[j+k]*inv2%MOD,a[i+j+k]=1ll*a[i+j+k]*inv2%MOD;
    }
}
```

0.7 Inv

```
long long ex_gcd(long long a,long long b,long long &x,long long &y)
{
    if(a==0&&b==0) return -1;
    if(b==0){x=1;y=0;return a;}
    long long d=ex_gcd(b,a%b,y,x);
    y-=a/b*x;
    return d;
}
long long inv(long long a,long long n)
{
    long long x,y;
    long long d=ex_gcd(a,n,x,y);
    if(d==1) return (x%n+n)%n;
    else return -1;
}
long long inv_(long long a,long long m)
{
    if(a==1) return 1;
    return inv_(m%a,m)*(m-m/a)%m;
}
long long inv_(long long a,long long mod){return pow(a,mod-2,mod);}
```

```
void inv_(){inv[0]=inv[1]=1;for(int i=2;i<MAX;++i) inv[i]=((MOD-MOD/i)*inv[MOD%i])%MOD
    ;}
```

0.8 Prim$_{pre}$

```
int euler(int n)
{
    int ans=n;
    for(int i=2;i*i<=n;i++)if(n%i==0)
    {
        ans-=ans/i;
        while(n%i==0) n/=i;
    }
    if(n>1) ans-=ans/n;
    return ans;
}
/* phi and prim*/
bool mark[MAX];int phi[MAX],prim[MAX],tot;
void phi_prim(int n)
{
    memset(mark,0,sizeof(mark));
    phi[1]=1;tot=0;
    for(int i=2;i<=n;i++)
    {
        if(!mark[i]){prim[++tot]=i;phi[i]=i-1;}
        for(int j=1;j<=tot;j++)
        {
            int x=prim[j];
            if(i*x>n) break;
            mark[i*x]=1;
            if(i%x==0){phi[i*x]=phi[i]*x;break;}
            else phi[i*x]=phi[i]*phi[x];
        }
    }
}
/* mo and du*/
const int MAX=1e7+5;
bool vis[MAX];int prim[MAX],mu[MAX],fac[MAX],tot,pcnt;
map<long long,long long>dp;
void moblus()
{
    mu[1]=1;tot=0;
    for(int i=2;i<MAX;i++)
    {
        if(!vis[i]) {prim[tot++]=i;mu[i]=-1;}
        for(int j=0;j<tot&&i*prim[j]<MAX;j++)
        {
            vis[i*prim[j]]=1;
            if(i%prim[j]) mu[i*prim[j]]=-mu[i];
            else {mu[i*prim[j]]=0;break;}
        }
    }
    for(int i=2;i<MAX;++i) mu[i]+=mu[i-1];//phi same
}
long long M(long long x)
{
    if(x<MAX) return mu[x];
    if(dp[x]) return dp[x];
```

```
    long long sum=1;//sum=0 phi->x*(x+1)/2;
    for(long long l=2,r;l<=x;l=r+1)
    {
        r=x/(x/l);
        sum-=M(x/l)*(r-l+1);
        //sum+=M(x/l)*(r-l+1)
    }
    return dp[x]=sum;//dp[x]=x*(x+1)/2-sum
}
```

0.9 Factor$_{D}ivide$

```
/* get n! divide :need pow.cpp prim.cpp */
int fac[MAX];
void factor(int a,int b)
{
    memset(fac,0,sizeof(fac));int i;
    for(i=1;i<=prim[0]&&prim[i]<=a;i++)
    {
        int tmp=a;
        while(tmp){fac[i]+=tmp/prim[i];tmp/=prim[i];}
    }
    fac[0]=i;
    for(i=1;i<=prim[0]&&prim[i]<=b;i++)
    {
        int tmp=b;
        while(tmp){fac[i]-=tmp/prim[i];tmp/=prim[i];}
    }
    for(i=1;i<=prim[0]&&prim[i]<=a-b;i++)
    {
        int tmp=a-b;
        while(tmp){fac[i]-=tmp/prim[i];tmp/=prim[i];}
    }
}
long long C(int a,int b)
{
    factor(a,b);long long c=1;
    for(int i=1;i<fac[0];i++) if(fac[i]) c=c*pow(prim[i],fac[i],MOD)%MOD;
    return c;
}
/* get n divide */
int fac[MAX][2],facnt;
int factor(long long x)
{
    memset(fac,0,sizeof(fac));facnt=0;
    long long tmp=x;
    for(int i=1;prim[i]<=tmp/prim[i];i++)
    {
        fac[facnt][1]=0;
        if(tmp%prim[i]==0)
        {
            fac[facnt][0]=prim[i];
            while(tmp%prim[i]==0){fac[facnt][1]++;tmp/=prim[i];}
            facnt++;
        }
    }
    if(tmp!=1){fac[facnt][0]=tmp;fac[facnt++][1]=1;}
    return facnt;
```

```
}
```

0.10 Lucas

```
long long n,m,MOD,fact[MAX];
inline long long inv_(long long a)
{
    if(a==1) return 1;
    return inv_(MOD%a)*(MOD-MOD/a)%MOD;
}
inline void init()
{
    fact[0]=1;
    for(int i=1;i<=MOD;i++) fact[i]=(fact[i-1]*i)%MOD;
}
inline long long C(long long a,long long b)
{
    if(b>a) return 0;
    return fact[a]*inv_(fact[a-b]*fact[b]%MOD)%MOD;
}
inline long long lucas(long long a,long long b)
{
    if(a<MOD&&b<MOD) return C(a,b);
    return lucas(a/MOD,b/MOD)*C(a%MOD,b%MOD)%MOD;
}
int main()
{
    int t;scanf("%d",&t);
    while(t--)
    {
        scanf("%I64d%I64d%I64d",&n,&m,&MOD);init();
        printf("%I64d\n",lucas(n+m,m));
    }
    return 0;
}
/* Chinese Left -MOD=m1*m2*m3...mx need pow.cpp*/
#include<bits/stdc++.h>
using namespace std;
const int MAX=1e5+5;
long long n,m,k,MOD,ans[MAX],m1[MAX];
inline long long C(long long a,long long b)
{
    if(b>a) return 0;
    long long x=1,y,z;
    for(int i=1;i<=b;++i)
    {
        y=(a+i-b)%MOD;
        z=i%MOD;
        x=x*(y*pow(z,MOD-2,MOD)%MOD)%MOD;
    }
    return x;
}
inline long long lucas(long long a,long long b)
{
    if(a<MOD&&b<MOD) return C(a,b);
    return lucas(a/MOD,b/MOD)*C(a%MOD,b%MOD)%MOD;
}
long long ex_gcd(long long a,long long b,long long &x,long long &y)
```

```
{
    if(b==0){x=1;y=0;return a;}
    long long d=ex_gcd(b,a%b,y,x);
    y-=a/b*x;
    return d;
}
long long muli(long long a,long long b,long long mod)
{
    a=(a%mod+mod)%mod;
    b=(b%mod+mod)%mod;
    long long ret=0;
    while(b)
    {
        if(b&1){ret+=a;if(ret>=mod)ret-=mod;}
        b>>=1;a<<=1;
        if(a>=mod) a-=mod;
    }
    return ret;
}
long long china()
{
    long long M=1,d,y,x=0;
    for(int i=0;i<k;++i) M*=m1[i];
    for(int i=0;i<k;++i)
    {
        long long w=M/m1[i];
        ex_gcd(m1[i],w,d,y);
        x=(x+muli(muli(y,w,M),ans[i],M));
    }
    return (x+M)%M;
}
int main()
{
    int t;scanf("%d",&t);
    while(t--)
    {
        scanf("%I64d%I64d%I64d",&n,&m,&k);
        for(int i=0;i<k;++i) scanf("%I64d",&m1[i]);
        for(int i=0;i<k;++i) {MOD=m1[i];ans[i]=lucas(n,m);}
        printf("%I64d\n",china());
    }
    return 0;
}
```

0.11 Polya

```
/* rotate */
for(i=1;i*i<n;++i)
{
    if(n%i) continue;
    ans+=euler(n/i)%p*pow(n%p,i-1,p)%p+euler(i)%p*pow(n%p,n/i-1,p)%p;
}
if(i*i==n) ans+=euler(i)%p*pow(n%p,i-1,p)%p;
/* rotate+Symmetric */
for(i=1;i*i<n;++i)
{
    if(n%i) continue;
    ans+=euler(n/i)*pow_(3,i)+euler(i)*pow_(3,n/i);
```

```
}
if(i*i==n) ans+=euler(i)*pow_(3,i);
ans/=n;
ans+=(pow_(3,(n+1)/2)+pow_(3,n/2+1))/2;
```

0.12 Gauss

```
/*x[]:ans      equ=var=n;*/
const double eps=1e-11;
double a[MAX][MAX],x[MAX];
int equ,var,n=20;
bool gauss()
{
    int i,j,k,col,max_r;
    for(k=0,col=0;k<equ&&col<var;++k,++col)
    {
        max_r=k;
        for(i=k+1;i<equ;++i) if(fabs(a[i][col])>fabs(a[max_r][col])) max_r=i;
        if(fabs(a[max_r][col])<eps) return 0;
        if(k!=max_r)
        {
            for(j=col;j<var;++j) swap(a[k][j],a[max_r][j]);
            swap(x[k],x[max_r]);
        }
        x[k]/=a[k][col];
        for(j=col+1;j<var;++j) a[k][j]/=a[k][col];
        a[k][col]=1.0;
        for(i=0;i<equ;++i) if(i!=k)
        {
            x[i]-=x[k]*a[i][col];
            for(j=col+1;j<var;++j) a[i][j]-=a[k][j]*a[i][col];
            a[i][col]=0.0;
        }
    }
    return 1;
}
```

0.13 Matrix$_{Tree}$

```
const int MAX=1e2+5;const long long MOD=1e9+7;
long long g[MAX][MAX];
void add(int x,int y)
{
    ++g[x][x];++g[y][y];
    --g[x][y];--g[y][x];
}
/*minimum tree*/
int n,m,p,g[MAX][MAX],vis[MAX],fa[MAX],ka[MAX];
long long ans,mat[MAX][MAX];
vector<int>gra[MAX];
struct P{int u,v,w;}e[MAX];
bool cmp(P ta,P tb){return ta.w<tb.w;}
long long gauss(int n)
{
    long long ans=1;
    for(int i=0;i<n;++i)
    {
```

```
19          for(int j=i+1;j<n;++j)
20          while(mat[j][i])
21          {
22              long long t=mat[i][i]/mat[j][i];
23              for(int k=i;k<n;++k) mat[i][k]=(mat[i][k]-t*mat[j][k]+p)%p;
24              swap(mat[i],mat[j]);
25              ans=-ans;
26          }
27          ans=(ans*mat[i][i])%p;
28          if(!ans) return 0;
29      }
30      return (ans+p)%p;
31  }
32  int find_(int x,int y[]) {return x==y[x]?x:find_(y[x],y);}
33  void matrix_tree()
34  {
35      for(int i=0;i<n;++i) if(vis[i]){gra[find_(i,ka)].push_back(i);vis[i]=0;}
36      for(int i=0;i<n;++i) if(gra[i].size()>1)
37      {
38          memset(mat,0,sizeof(mat));
39          int len=gra[i].size();
40          for(int j=0;j<len;++j)
41          for(int k=j+1;k<len;++k)
42          {
43              int u=gra[i][j],v=gra[i][k];
44              if(g[u][v])
45              {
46                  mat[k][j]=(mat[j][k]-=g[u][v]);
47                  mat[k][k]+=g[u][v];mat[j][j]+=g[u][v];
48              }
49          }
50          ans=ans*gauss(gra[i].size()-1)%p;
51          for(int j=0;j<len;++j) fa[gra[i][j]]=i;
52      }
53      for(int i=0;i<n;++i) {gra[i].clear();ka[i]=fa[i]=find_(i,fa);}
54  }
55  int main()
56  {
57      while(~scanf("%d%d%d",&n,&m,&p))
58      {
59          if(n==0&&m==0&&p==0) break;
60          memset(g,0,sizeof(g));ans=1;
61          for(int i=0;i<m;++i) {scanf("%d%d%d",&e[i].u,&e[i].v,&e[i].w);--e[i].u;--e[i].v
        ;}
62          sort(e,e+m,cmp);
63          for(int i=0;i<n;++i) ka[i]=fa[i]=i;
64          for(int i=0;i<=m;++i)
65          {
66              if((i&&e[i].w!=e[i-1].w)||i==m) matrix_tree();
67              long long u=find_(e[i].u,fa),v=find_(e[i].v,fa);
68              if(u!=v)
69              {
70                  vis[v]=vis[u]=1;
71                  ka[find_(u,ka)]=find_(v,ka);
72                  ++g[u][v],++g[v][u];
73              }
74          }
75          int flag=1;
76          for(int i=1;i<n;++i) if(fa[i]!=fa[i-1]) flag=0;
```

```
77          printf("%lld\n",flag?ans%p:0);
78      }
79      return 0;
80  }
```

1 String Processing

1.1 Z-Box

```
/*
z[i]:string'sa[i] compare with string the length-LCP
S=P+$+T
z[strlen(P)+1~~strlen(P)+strlen(T)]==length(T)--'s idex -strlen(P)-1:the place P appear in T
S=T+$+P
the number of z[strlen(T)+1~~strlen(T)+strlen(P)]!=0:the number of T'sa[i] is P'prefix
*/
void z-box()
{
    z[0]=n;
    for (int i=1,j=1,k;i<n;i=k)
    {
        if (j<i) j=i;
        while (j<n&&S[j]==S[j-i]) ++j;
        z[i]=j-i;k=i+1;
        while (k+z[k-i]<j) z[k]=z[k-i],++k;
    }
}
```

1.2 Hash

```
/*Longest Palindrome string*/
const long long P=131;
long long power[MAX],ha1[MAX],ha2[MAX];
bool check(int l1,int r1,int l2,int r2)
{
    long long tmp1=ha1[r1]-ha1[l1-1]*power[r1-l1+1];
    long long tmp2=ha2[r2]-ha2[l2+1]*power[l2+1-r2];
    return tmp1==tmp2;
}
int main()
{
    power[0]=1;for(int i=1;i<MAX-1;++i) power[i]=power[i-1]*P;
    L=strlen(s+1);
    ha1[0]=ha2[L+1]=0;
    for(int i=1;i<=L;++i) ha1[i]=ha1[i-1]*P+s[i]-'a';
    for(int i=L;i>=1;--i) ha2[i]=ha2[i+1]*P+s[i]-'a';
    while(l<=r){mid;if(check(i-mid,i-1,i+mid,i+1));else;}// fen ji&ou
}
/*multi hash */
const int HASH=10;
int AC[HASH]={131,137,139,149,151,157,163,167,173,179};
int ACC[HASH]={200003,200009,200017,200023,200029,200033,200041,200063,200087,200117};
long long bas[MAX][HASH],sum1[MAX][HASH];
bool check(int index,int x,int len)
{
    long long ha1=((sum1[x+len-1][index]-sum1[x-1][index]*bas[len][index]%ACC[index])%ACC[index]+ACC[index])%ACC[index];
    long long ha2=((sum1[n-x+1][index]-sum1[n-x-len+1][index]*bas[len][index]%ACC[index])%ACC[index]+ACC[index])%ACC[index];
    if(ha1==ha2) return true;
    else return false;
}
```

```
bool check(int x,int len)
{
    for(int i=0;i<HASH;i++) if(!check(i,x,len)) return false;
    return true;
}
int main()
{
    for (int i=0;i<HASH;i++)
    {
        bas[0][i]=1;
        for(int j=1;j<=n;j++) bas[j][i]=bas[j-1][i]*AC[i]%ACC[i];
        for(int j=1;j<=n;j++) sum1[j][i]=(sum1[j-1][i]*AC[i]%ACC[i]+s[j]-'a'+1)%ACC[i];
    }
    return 0;
}
```

1.3 AC

```
/*AC->build Tire ->build Mat or Dp*/
struct Trie
{
    static const int MAXN=26;//MAXN will change
    int nxt[MAX][MAXN],f[MAX],e[MAX],rt,L;
    int newnode()
    {
        for(int i=0;i<MAXN;++i) nxt[L][i]=-1;
        e[L++]=0;return L-1;
    }
    void init(){L=0;rt=newnode();}
    void insert(char *buf)
    {
        int len=strlen(buf),now=rt;
        for(int i=0;i<len;++i)
        {
            int x=buf[i]-'a';
            if(nxt[now][x]==-1) nxt[now][x]=newnode();
            now=nxt[now][x];
        }
        ++e[now];//e[now]=1;e[now]=id;//e[now]=1<<id;
    }
    void build()
    {
        queue<int>q;f[rt]=rt;
        for(int i=0;i<MAXN;++i)
        if(nxt[rt][i]==-1) nxt[rt][i]=rt;
        else
        {
            f[nxt[rt][i]]=rt;
            q.push(nxt[rt][i]);
        }
        while(!q.empty())
        {
            int now=q.front();q.pop();
            for(int i=0;i<MAXN;++i)
            if(nxt[now][i]==-1) nxt[now][i]=nxt[f[now]][i];
            else
            {
                f[nxt[now][i]]=nxt[f[now]][i];
```

```
41                  q.push(nxt[now][i]);
42                  //e[nxt[now][i]] op e[f[nxt[now][i]]]; | or + or =
43              }
44          }
45      }
46      int query(char *buf,other..)
47      {
48          int len=strlen(buf),now=rt,res=0;
49          for(int i=0;i<len;++i)
50          {
51              now=nxt[now][buf[i]-'a'];int tmp=now;
52              while(tmp!=rt){res+=e[tmp];e[tmp]=0;tmp=f[tmp];}
53          }
54          return res;
55      }
56      void get(Mat &a)
57      {
58          for(int i=0;i<L;++i)if(!e[i])
59          for(int j=0;j<MAXN;++j)if(!e[nxt[i][j]])
60          a.m[i][nxt[i][j]]=(a.m[i][nxt[i][j]]+1)%MOD;
61      }
62      void spfa(int k)//need dis[] and g[][] and pos[cnt++]
63      {
64          queue<int>q;memset(dis,-1,sizeof(dis));dis[pos[k]]=0;
65          q.push(pos[k]);
66          while(!q.empty())
67          {
68              int now=q.front();q.pop();
69              for(int i=0;i<2;++i)
70              {
71                  int tmp=nxt[now][i];
72                  if(dis[tmp]<0&&e[tmp]>=0) {dis[tmp]=dis[now]+1;q.push(tmp);}
73              }
74          }
75          for(int i=0;i<cnt;++i) g[k][i]=dis[pos[i]];
76      }
77  }AC;
78  int main()
79  {
80      AC.init();for(){AC.insert(char []);}AC.build();
81      /* get g[][] and then dp find a shortest path */
82      pos[0]=0;cnt=1;
83      for(int i=0;i<AC.L;++i)if(AC.e[i]>0) pos[cnt++]=i;
84      for(int i=0;i<cnt;++i) AC.spfa(i);
85      /* dp */
86      int xi=AC.nxt[i][j],xhash=hash|?;//save the state
87      dp[xi][xhash]=max(dp[xi][xhash],dp[i][hash]+AC.e[xi]);
88      for(int i=0;i<AC.L;++i) ans=max(ans,dp[i][hash]);
89      /*if resort the string and find a special string,please be careful about the cnt[
        char] can be max like that */
90      for(int j=0;j<AC.MAXN;++j) {if(j==0&&a==cnt[0]) continue;}
91      /*less val and less string */
92      strcpy(str[0][0],"");strcpy(ans,"");
93      int max_=0;
94      for(int i=0;i<n;++i)
95      for(int j=0;j<AC.L;++j)if(dp[i][j]>=0)
96      {
97          strcpy(tmp,str[i][j]);
98          int len=strlen(tmp);
```

```
        for(int k=0;k<AC.MAXN;++k)
        {
            int xi=i+1,xj=AC.nxt[j][k],t=dp[i][j];
            tmp[len]='a'+k;tmp[len+1]='\0';
            if(AC.e[xj]) t+=val[AC.e[xj]];
            if(dp[xi][xj]<t||(dp[xi][xj]==t&&cmp(tmp,str[xi][xj])))
            {
                dp[xi][xj]=t;
                strcpy(str[xi][xj],tmp);
                if(max_<t||(max_==t&&cmp(tmp,ans))){max_=t;strcpy(ans,tmp);}
            }
        }
    }
}
/* if need shortest minimum order string */
bool cmp(char *a,char *b)
{
    int al=strlen(a),bl=strlen(b);
    return al==bl?strcmp(a,b)<0:al<bl;
}
```

1.4 PT

```
/*
PT.L-2:The Number different of Palindromes
num[i]:The Number different of Palindromes in Palindromes String i
cnt[i]:The Number appear of Palindromes String i
*/
struct Trie
{
    static const int MAXN=26;
    int nxt[MAX][MAXN],f[MAX],cnt[MAX],num[MAX],len[MAX],c[MAX],last,n,L;
    int newnode(int x)
    {
        for(int i=0;i<MAXN;++i) nxt[L][i]=0;
        cnt[L]=num[L]=0;len[L]=x;return L++;
    }
    void init()
    {
        L=0;newnode(0);newnode(-1);
        last=0;n=0;c[n]=-1;f[0]=1;
    }
    int getf(int x)
    {
        while(c[n-len[x]-1]!=c[n]) x=f[x];
        return x;
    }
    void add(int x)
    {
        x-='a';c[++n]=x;
        int cur=getf(last);
        if(!nxt[cur][x])
        {
            int now=newnode(len[cur]+2);
            f[now]=nxt[getf(f[cur])][x];
            nxt[cur][x]=now;
            num[now]=num[f[now]]+1;
        }
```

```
        ++cnt[last=nxt[cur][x]];
    }
    void count(){for(int i=L-1;i>=2;--i) cnt[f[i]]+=cnt[i];}
}PT;
int main()
{
    PT.init();for(int i=0;i<l;++i) PT.add(char i);PT.count();
    for(int i=2;i<PT.L;++i) //??
    return 0;
}
/*the number of two string' same Palindromes pairs*/
long long dfs(int u,int v)
{
    long long tmp=0;
    for(int i=0;i<26;++i) if(PT1.nxt[u][i]&&PT2.nxt[v][i])
    {
        tmp+=1ll*PT1.cnt[PT1.nxt[u][i]]*PT2.cnt[PT2.nxt[v][i]];
        tmp+=dfs(PT1.nxt[u][i],PT2.nxt[v][i]);
    }
    return tmp;
}
```

1.5 SAM

```
/*2 string LCA and multi string LCA */
struct Tire
{
    static const int MAXN=26;
    int nxt[MAX][MAXN],f[MAX],L[MAX],last,tot;
    void init()
    {
        last=tot=0;memset(nxt[0],-1,sizeof(nxt[0]));
        f[0]=-1;L[0]=0;
    }
    void add(int x)
    {
        int p=last,np=++tot;L[np]=L[p]+1;
        memset(nxt[np],-1,sizeof(nxt[np]));
        while(~p&&nxt[p][x]==-1) nxt[p][x]=np,p=f[p];
        if(p==-1) f[np]=0;
        else
        {
            int q=nxt[p][x];
            if(L[q]!=L[p]+1)
            {
                int nq=++tot;
                L[nq]=L[p]+1;
                memcpy(nxt[nq],nxt[q],sizeof(nxt[q]));
                f[nq]=f[q]; f[q]=f[np]=nq;
                while(~p&&nxt[p][x]==q) nxt[p][x]=nq,p=f[p];
            }
            else f[np]=q;
        }
        last=np;
    }
    int find(char *s)
    {
        int len=strlen(s);
```

```
35          int res=0,tmp=0,u=0;
36          for(int i=0;i<len;++i)
37          {
38              int x=s[i]-'a';
39              if(~nxt[u][x]) ++tmp,u=nxt[u][x];
40              else
41              {
42                  while(~u&&nxt[u][x]==-1) u=f[u];
43                  if(~u) tmp=L[u]+1,u=nxt[u][x];
44                  else tmp=0,u=0;
45              }
46              res=max(res,tmp);
47              //Max[u]=max(Max[u],tmp);//multi string
48          }
49          //for(int i=tot;i>=1;--i) Max[f[i]]=max(Max[f[i]],Max[i]);//multi string
50          //for(int i=0;i<=tot;++i) Min[i]=min(Min[i],Max[i]);//multi string
51          return res;
52      }
53      /*
54      int cal()//multi string
55      {
56          int res=0;
57          for(int i=0;i<=tot;++i) res=max(res,Min[i]);
58          return res;
59      }
60      */
61      void cal()//topsort
62      {
63          memset(in,0,sizeof(in));
64          for(int i=1;i<=tot;++i) ++in[f[i]];
65          queue<int >q;
66          for(int i=1;i<=tot;++i) if(!in[i]) q.push(i);
67          while(!q.empty())
68          {
69              int u=q.front();q.pop();
70              if(f[u]==-1) continue;
71              rt[f[u]]+=rt[u];
72              if(--in[f[u]]==0) q.push(f[u]);
73          }
74          memset(ans,0,sizeof(ans));
75          for(int i=1;i<=tot;++i) ans[L[i]]=max(ans[L[i]],rt[i]);//the max number of
        length L[i] strings
76      }
77      /*int who[maxn], a[maxn];*/
78      void sort()
79      {
80          for(int i=1;i<=tot;i++) a[i]=0;
81          for(int i=1;i<=tot;i++) a[L[i]]++;
82          for(int i=1;i<=tot;i++) a[i]+=a[i-1];
83          for(int i=1;i<=tot;i++) who[a[L[i]]--]=i;
84          /* dp */
85          for(int i=tot;i>=1;i--)
86          {
87              long long sum=0;int p=who[i];
88              for(int j=0;j<26;j++) if(~nxt[p][j]) sum+=dp[nxt[p][j]];
89              dp[p]=sum+1;
90          }
91      }
92      */
```

```
}SAM;
int main()
{
    SAM.init();for(int i=0;i<len;++i) SAM.add(s[i]-'a');
    printf("%d\n",SAM.find(s));
    /*the minimum represent:S+S:L[now]-len+1; */
    for(int i = 0; i < len; i++)
    for(int j = 0; j < 26; j++) if(nxt[now][j] != NULL){now=nxt[now][j];break;}
    return 0;
}
/*L[i]-L[f[i]]:number of different substrings*/
/*the minimum represent*/
int minrepresent(char *s)
{
    int i=0,j=1,k=0;
    int len=strlen(s);
    while(i<len&&j<len&&k<len)
    {
        if(s[(i+k)%len]==s[(j+k)%len]) k++;
        else
        {
            if(s[(i+k)%len]>s[(j+k)%len]) i=i+k+1; else j=j+k+1;
            if(i==j) ++j; k=0;
        }
    }
    return i<j?i:j;
}
```

2 Data Structure

2.1 Bit$_{T}ree$

```
/*point update region query*/
void add(int x,int v){while(x<=n) c[x]+=v,x+=x&(-x);}
int get(int r)
{
    int sum=0;
    while(r) sum+=c[r],r-=r&(-r);
    return sum;
}
int get(int l, int r){return get(r)-get(l-1);}
/*point update region query*/
void add(int r, int v){while(r<=n) c[r]+=v,r+=r&(-r);}
void add(int l,int r,int x){add(l,x),add(r+1,-x);}
int get(int x)
{
    int sum=0;
    while(x) sum+=c[x],x-=x&(-x);
    return sum;
}
/* region update region query*/
void add(int r,int v){for(int i=r;i<=n;i+=i&(-i)){c1[i]+=v;c2[i]+=r*v;}}
void add(int l,int r,int v){add(l,v);add(r+1,-v);}
int get(int r)
{
    int sum=0;
    for(int i=r;i>0;i-=i&(-i)) sum+=(r+1)*c1[i]-c2[i];
    return sum;
}
int get(int l,int r){return get(r)-get(l-1);}
```

2.2 Block

```
const int eps=1e-8;
int a[MAX],aa[MAX],b[MAX],c[MAX],cc[MAX][2],block,cnt;
void sort(int x)
{
    int L=cc[x][0],R=cc[x][1];
    for(int k=L;k<=R;k++) aa[k]=a[k];
    sort(aa+L,aa+R+1);
}
void add(int l,int r,int w)
{
    bool bl=l==cc[c[l]][0],br=r==cc[c[r]][1];
    if(c[l]==c[r]){for(int k=l;k<=r;k++) a[k]+=w; sort(c[l]);}
    else
    {
        if(!bl){for(int k=l;k<=cc[c[l]][1];k++) a[k]+=w; sort(c[l]);}
        if(!br){for(int k=cc[c[r]][0];k<=r;k++) a[k]+=w; sort(c[r]);}
        for(int k=c[l]+1-bl;k<c[r]+br;k++) b[k]+=w;
    }
}
int query(int l,int r,int w)
{
    int ans=0;
    bool bl=l==cc[c[l]][0],br=r==cc[c[r]][1];
```

```
    if(c[l]==c[r]){for(int k=l,v=w-b[c[l]];k<=r;k++) ans+=a[k]>=v;}
    else
    {
        if(!bl)for(int k=l,v=w-b[c[l]];k<=cc[c[l]][1];k++) ans+=a[k]>=v;
        if(!br)for(int k=cc[c[r]][0],v=w-b[c[r]];k<=r;k++) ans+=a[k]>=v;
        for(int k=c[l]+1-bl;k<c[r]+br;k++)
        {
            int L=cc[k][0],R=cc[k][1],v=w-b[k],mid;
            while(L<=R)
            {
                mid=(L+R)>>1;
                aa[mid]>=v?R=mid-1:L=mid+1;
            }
            ans+=cc[k][1]-L+1;
        }
    }
    return ans;
}
int main()
{
    block=(int)(sqrt(n)+eps),cnt=1;
    for(int k=1;k<=n;k++) c[k]=k%block==0?cnt++:cnt;
    for(int k=1;k<=cnt;k++) {cc[k][0]=k*block-block+1;cc[k][1]=k*block;}cc[cnt][1]=n;
    for(int k=1;k<=cnt;k++) sort(k);
    add(x,y,z);query(x,y,z);
    return 0;
}
```

2.3 $\mathbf{Line}_{T}ree_{U}nion$

```
int ls[MAX<<6],rs[MAX<<6],val[MAX<<6],d[MAX],rt[MAX],cnt;
void update(int &x,int l,int r,int p,int v)
{
    x=++cnt;if(l==r){val[x]=v;return;}
    int mid=(l+r)>>1;
    if(p<=mid) update(ls[x],l,mid,p,v);
    else update(rs[x],mid+1,r,p,v);
    val[x]=min(val[ls[x]],val[rs[x]]);
}
int merge_(int u,int v)
{
    if(!u) return v;
    if(!v) return u;
    int x=++cnt;
    ls[x]=merge_(ls[u],ls[v]);
    rs[x]=merge_(rs[u],rs[v]);
    if(ls[x]||rs[x]) val[x]=min(val[ls[x]],val[rs[x]]);
    else val[x]=min(val[u],val[v]);
    return x;
}
int query(int x,int l,int r,int L,int R)
{
    if(!x) return INF;
    if(l==L&&r==R) return val[x];
    int mid=(l+r)>>1;
    if(R<=mid) return query(ls[x],l,mid,L,R);
    if(L>mid) return query(rs[x],mid+1,r,L,R);
    return min(query(ls[x],l,mid,L,mid),query(rs[x],mid+1,r,mid+1,R));
```

```
}
void dfs(int u,int fa)
{
    update(rt[u],1,100000,d[u],a[u]);
    for(int i=head[u];~i;i=e[i].nxt)
    {
        int v=e[i].to;
        if(v==fa) continue;
        d[v]=d[u]+1;
        dfs(v,u);
        rt[u]=merge_(rt[u],rt[v]);
    }
}
```

2.4 Tree$_{C}ut$

```
struct P{int to,nxt;}e[MAX<<1];
struct Px{int l,r,v;Px(){l=r=-1;v=0;}void is(int a1,int a2,int a3){l=a1;r=a2;v=a3;}}b[
    MAX<<2];
int head[MAX],top[MAX],fa[MAX],deep[MAX],num[MAX],p[MAX],fp[MAX],son[MAX],tot,pos,n,m,a
    [MAX],lz[MAX<<2];
void init()
{
    tot=0;pos=1;
    memset(head,-1,sizeof(head));
    memset(son,-1,sizeof(son));
    memset(lz,-1,sizeof(lz));
}
void adde(int u,int v)
{
    e[tot].to=v;e[tot].nxt=head[u];head[u]=tot++;
    e[tot].to=u;e[tot].nxt=head[v];head[v]=tot++;
}
void dfs(int u,int pre,int d)
{
    deep[u]=d;fa[u]=pre;num[u]=1;
    for(int i=head[u];i!=-1;i=e[i].nxt)
    {
        int v=e[i].to;
        if(v!=pre)
        {
            dfs(v,u,d+1);
            num[u]+=num[v];
            if(son[u]==-1||num[v]>num[son[u]]) son[u]=v;
        }
    }
}
void getpos(int u,int sp)
{
    top[u]=sp;p[u]=pos++;fp[p[u]]=u;
    if(son[u]==-1) return ;
    getpos(son[u],sp);
    for(int i=head[u];i!=-1;i=e[i].nxt)
    {
        int v=e[i].to;
        if(v!=son[u]&&v!=fa[u]) getpos(v,v);
    }
}
```

```
inline Px merge_(const Px &x,const Px &y)//may not need
{
    if(x.v==0) return y;
    if(y.v==0) return x;
    Px t;
    t.is(x.l,y.r,x.v+y.v);
    if(x.r==y.l) --t.v;
    return t;
}
inline void pushup(int rt)
{
    b[rt].is(b[rt<<1].l,b[rt<<1|1].r,b[rt<<1].v+b[rt<<1|1].v);
    if(b[rt<<1].r==b[rt<<1|1].l) --b[rt].v;
}
inline void pushdown(int rt)
{
    b[rt<<1].is(lz[rt],lz[rt],1);
    b[rt<<1|1].is(lz[rt],lz[rt],1);
    lz[rt<<1]=lz[rt<<1|1]=lz[rt];
    lz[rt]=-1;
}
void build(int rt,int l,int r)
{
    if(l==r) {b[rt].is(a[fp[r]],a[fp[r]],1);return ;}
    int mid=(l+r)>>1;
    build(rt<<1,l,mid);
    build(rt<<1|1,mid+1,r);
    pushup(rt);
}
void update(int rt,int l,int r,int L,int R,int v)
{
    if(L<=l&&R>=r) {b[rt].is(v,v,1);lz[rt]=v;return ;}
    if(lz[rt]!=-1) pushdown(rt);
    int mid=(l+r)>>1;
    if(L<=mid) update(rt<<1,l,mid,L,R,v);
    if(R>mid) update(rt<<1|1,mid+1,r,L,R,v);
    pushup(rt);
}
Px query(int rt,int l,int r,int L,int R)
{
    if(L<=l&&R>=r) return b[rt];
    if(lz[rt]!=-1) pushdown(rt);
    int mid=(l+r)>>1;Px t;
    if(L<=mid) t=query(rt<<1,l,mid,L,R);
    if(R>mid) t=merge_(t,query(rt<<1|1,mid+1,r,L,R));
    return t;
}
void update(int l,int r,int v)
{
    while(top[l]!=top[r])
    {
        if(deep[top[l]]<deep[top[r]]) swap(l,r);
        update(1,1,n,p[top[l]],p[l],v);
        l=fa[top[l]];
    }
    if(deep[l]>deep[r]) swap(l,r);
    update(1,1,n,p[l],p[r],v);
}
int query(int l,int r)
```

```
{
    Px L,R;//*
    while(top[l]!=top[r])
    {
        if(deep[top[l]]<deep[top[r]]) {swap(l,r);swap(L,R);}
        L=merge_(query(1,1,n,p[top[l]],p[l]),L);//*
        l=fa[top[l]];
    }
    if(deep[l]>deep[r]) {swap(l,r);swap(L,R);}
    R=merge_(query(1,1,n,p[l],p[r]),R);//*
    return L.v+R.v-(L.l==R.l?1:0);
}
int main()
{
    while(~scanf("%d%d",&n,&m))
    {
        init();
        dfs(1,0,0);getpos(1,1);
        build(1,1,n);
    }
    return 0;
}
```

2.5 President$_{T}ree$

```
/the k-max number of region */
int root[MAX],a[MAX],n,m,cnt;
struct P{int l,r,v;}b[MAX*25];
vector<int >v;
int id(int x){return lower_bound(v.begin(),v.end(),x)-v.begin()+1;}
void update(int l,int r,int &x,int y,int pos)
{
    b[++cnt]=b[y];++b[cnt].v;x=cnt;
    if(l==r) return ;
    int mid=(l+r)>>1;
    if(pos<=mid) update(l,mid,b[x].l,b[y].l,pos);
    else update(mid+1,r,b[x].r,b[y].r,pos);
}
int query(int l,int r,int x,int y,int k)
{
    if(l==r) return l;
    int mid=(l+r)>>1;
    int sum=b[b[y].l].v-b[b[x].l].v;
    if(sum>=k) return query(l,mid,b[x].l,b[y].l,k);
    else return query(mid+1,r,b[x].r,b[y].r,k-sum);
}
int main()
{
    while(~scanf("%d%d",&n,&m))
    {
        cnt=0;v.clear();
        for(int i=1;i<=n;++i) scanf("%d",&a[i]),v.push_back(a[i]);
        sort(v.begin(),v.end());
        v.erase(unique(v.begin(),v.end()),v.end());
        for(int i=1;i<=n;++i) update(1,n,root[i],root[i-1],id(a[i]));
        for(int i=1;i<=m;++i)
        {
            scanf("%d%d%d",&x,&y,&k);
```

```
34              printf("%d\n",v[query(1,n,root[x-1],root[y],k)-1]);
35          }
36      }
37      return 0;
38  }
```

3 Graph Theory

3.1 $Bipartite_{G}raph_{M}atching$

```
/*one Matching */
const int MAX1=5*1e3+1;
int linker[MAX1],n;
bool used[MAX1];
bool dfs(int u)
{
    for (int i=head[u];i!=-1;i=edge[i].next)
    {
        int v=edge[i].to;
        if(!used[v])
        {
            used[v]=1;
            if(linker[v]==-1||dfs(linker[v])){linker[v]=u;return 1;}
        }
    }
    return 0;
}
int hungary()
{
    int res=0;memset(linker,-1,sizeof(linker));
    for(int u=1;u<=n;u++)
    {
        memset(used,0,sizeof(used));
        if(dfs(u)) res++;//return 0;
    }
    return res;//return 1;
}
/* multi Matching */
int lg[MAX1][MAX2],inker[MAX2][MAX1],vlink[MAX2],num[MAX2];bool used[MAX2];
bool dfs(int u)
{
    for(int v=1;v<=m;v++) if(g[u][v]&&!used[v])
    {
        used[v]=1;
        if(vlink[v]<num[v]) {linker[v][++vlink[v]]=u;return 1;}
        for(int k=1;k<=vlink[v];k++) if(dfs(linker[v][k]))
        {
            linker[v][k]=u;
            return 1;
        }
    }
    return 0;
}
int hungary()
{
    memset(linker,-1,sizeof(linker));memset(vlink,0,sizeof(vlink));
    for(int u=1;u<=n;u++)
    {
        memset(used,0,sizeof(used));
        if(!dfs(u)) return 0;
    }
    return 1;
}
/* max val Matching -KM*/
int linker[MAX],lx[MAX],ly[MAX],slack[MAX];
```

```
int visx[MAX],visy[MAX],w[MAX][MAX];
int dfs(int x)
{
    visx[x]=1;
    for(int y=1;y<=ny;y++)
    {
        if(visy[y]) continue;
        int tmp=lx[x]+ly[y]-w[x][y];
        if(tmp==0)
        {
            visy[y]=1;
            if(linker[y]==-1||dfs(linker[y])) {linker[y]=x;return 1;}
        }
        else if(slack[y]>tmp) slack[y]=tmp;
    }
    return 0;
}
int km()
{
    int i,j;
    memset(linker,-1,sizeof(linker)); memset(ly,0,sizeof(ly));
    for(i=1;i<=nx;i++)
    for(j=1,lx[i]=-INF;j<=ny;j++) if(w[i][j]>lx[i]) lx[i]=w[i][j];
    for(int x=1;x<=nx;x++)
    {
        for(i=1;i<=ny;i++) slack[i]=INF;
        while(1)
        {
            memset(visx,0,sizeof(visx)); memset(visy,0,sizeof(visy));
            if(dfs(x)) break;
            int d=INF;
            for(i=1;i<=ny;i++) if(!visy[i] && d>slack[i]) d=slack[i];
            for(i=1;i<=nx;i++) if(visx[i]) lx[i]-=d;
            for(i=1;i<=ny;i++) if(visy[i]) ly[i]+=d; else slack[i]-=d;
        }
    }
    int res=0;
    for(i=1;i<=ny;i++) if(linker[i]!=-1) res+=w[linker[i]][i];
    return res;
}
int main()
{
    int n,m;char c;int top1,top2;pair<int,int>a[MAX],b[MAX];
    while(~scanf("%d%d",&n,&m)&&n&&m)
    {
        top1=top2=0;memset(w,0,sizeof(w));
        for(int k=0;k<n;k++)
        {
            c=getchar();
            for(int i=0;i<m;i++)
            {
                c=getchar();
                if(c=='H') a[++top1]=make_pair(k,i);
                else if(c=='m') b[++top2]=make_pair(k,i);
            }
        }
        for(int k=1;k<=top1;k++)
        for(int i=1;i<=top2;i++) w[k][i]=-abs(a[k].x-b[i].x)-abs(a[k].y-b[i].y);
        nx=top1;ny=top2;
```

```
        printf("%d\n",-km());
    }
    return 0;
}
```

3.2 Point$_D ivide$

```
/* get the number pair of point which dis<c */
struct P{int to,nxt,v;void is(int x1,int x2,int x3){to=x1;nxt=x2;v=x3;}}e[MAX<<1];
int head[MAX],sz[MAX],dis[MAX],maxp[MAX],rem[MAX],cnt,sum,rt,ans;
bool vis[MAX];
void init()
{
    cnt=ans=rt=0;rem[0]=0;
    memset(head,-1,sizeof(head));
    memset(vis,0,sizeof(vis));
}
void adde(int u,int v,int w)
{
    e[cnt].is(v,head[u],w);head[u]=cnt++;
    e[cnt].is(u,head[v],w);head[v]=cnt++;
}
void getrt(int u,int pa)
{
    sz[u]=1; maxp[u]=0;
    for(int i=head[u];i!=-1;i=e[i].nxt)
    {
        int to=e[i].to;
        if(to==pa||vis[to]) continue;
        getrt(to,u);
        sz[u]+=sz[to];//update the number of subnode in root [u],[to] is [u]'s subnode
        maxp[u]=max(maxp[u],sz[to]);//update the max of root [u],if sz [to] is greater
    }
    maxp[u]=max(maxp[u],sum-sz[u]);//in_ex
    if(maxp[u]<maxp[rt]) rt=u;
}
void getdis(int u,int fa)
{
    rem[++rem[0]]=dis[u];
    for(int i=head[u];i!=-1;i=e[i].nxt)
    {
        int to=e[i].to;
        if(to==fa||vis[to])continue;
        dis[to]=dis[u]+e[i].v;
        getdis(to,u);
    }
}
int cal(int u,int ct)
{
    dis[u]=ct;rem[0]=0;
    getdis(u,0);
    sort(rem+1,rem+rem[0]+1);
    int l=1,r=rem[0],x=0;
    while(l<r) if(rem[l]+rem[r]<=m){x+=r-l;++l;} else --r;
    //=c:for(int l=1;l<r;++l){while(rem[l]+rem[r]==m) ++x;--r;}
    return x;
}
int divide(int u)
```

```
52  {
53      vis[u]=1;ans+=cal(u,0);
54      for(int i=head[u];i!=-1;i=e[i].nxt)//divide the subtree
55      {
56          int to=e[i].to;
57          if(vis[to])continue;
58          ans-=cal(to,e[i].v);
59          sum=sz[to];rt=0;
60          getrt(to,0); divide(rt);
61      }
62      return ans;
63  }
64  int main()
65  {
66      while(~scanf("%d%d",&n,&m))
67      {
68          if(n==0&&m==0) break;init();
69          for(int i=1;i<n;++i){scanf("%d%d%d",&x,&y,&z);adde(x,y,z);}
70          maxp[rt]=sum=n;getrt(1,0);
71          printf("%d\n",divide(rt));
72      }
73      return 0;
74  }
```

4 Computational Geometry

4.1 Scanning$_{line}$

```
/* Overlap area */
const int MAX=2*1e3+1e2;
const double eps=1e-10;
struct node{double l,r,y;int v;}edge[MAX<<1];
double hash_[MAX<<1],one[MAX<<2],two[MAX<<2];int mark[MAX<<2],cnt,n;
inline bool cmp(const node &a,const node &b){return a.y-b.y<-eps;}
inline int find_(const double &x)
{
    int l=1,r=cnt,mid;
    while(l<=r)
    {
        mid=(l+r)>>1;
        if(hash_[mid]-x<-eps) l=mid+1;
        else if(hash_[mid]-x>eps) r=mid-1;
        else break;
    }
    return mid;
}
void pushup(int rt,int l,int r)
{
    if(mark[rt]==0){one[rt]=one[rt<<1]+one[rt<<1|1];two[rt]=two[rt<<1]+two[rt<<1|1];}
    else if(mark[rt]==1){one[rt]=hash_[r+1]-hash_[l];two[rt]=one[rt<<1]+one[rt<<1|1];}
    else if(mark[rt]>=2){one[rt]=two[rt]=hash_[r+1]-hash_[l];}
    /* Union
    if(mark[rt]) sum[rt]=hash[r+1]-hash[l];
    else if(l==r) sum[rt]=0;
    else sum[rt]=sum[rt<<1]+sum[rt<<1|1];*/
}
void update(int rt,int l,int r,int L,int R,int v)
{
    if(L<=l&&R>=r) {mark[rt]+=v;pushup(rt,l,r);return;}
    int mid=(l+r)>>1;
    if(L<=mid) update(rt<<1,l,mid,L,R,v);
    if(R>mid) update(rt<<1|1,mid+1,r,L,R,v);
    pushup(rt,l,r);
}
int main()
{
    int t;double a,b,c,d;scanf("%d",&t);
    while(t--)
    {
        double ans=0;memset(one,0,sizeof(one));memset(two,0,sizeof(two));memset(mark,0,
    sizeof(mark));
        scanf("%d",&n);
        for(int i=1;i<=n;i++)
        {
            scanf("%lf%lf%lf%lf",&a,&b,&c,&d);
            edge[i*2-1].l=edge[i*2].l=a;
            edge[i*2-1].r=edge[i*2].r=c;
            edge[i*2-1].y=b;edge[i*2].y=d;
            edge[i*2-1].v=1;edge[i*2].v=-1;
            hash_[i*2-1]=a;hash_[i*2]=c;
        }
        sort(hash_+1,hash_+n*2+1);
        cnt=unique(hash_+1,hash_+n*2+1)-hash_-1;
```

```
55          sort(edge+1,edge+n*2+1,cmp);
56          for(int i=1;i<n*2;i++)
57          {
58              int L=find_(edge[i].l),R=find_(edge[i].r)-1;
59              update(1,1,cnt,L,R,edge[i].v);
60              ans+=two[1]*(edge[i+1].y-edge[i].y);
61          }
62          printf("%.2f\n",ans);
63      }
64      return 0;
65  }
```

4.2 Convex$_hull$

```
1   const double eps=1e-8;
2   struct point
3   {
4       double x,y;
5       point(){}
6       point(double a,double b){x=a;y=b;}
7       point operator -(const point &a)const{return point(x-a.x,y-a.y);}
8       double operator ^(const point &a)const{return x*a.y-y*a.x;}
9       double operator *(const point &a)const{return x*a.x+y*a.y;}
10  }p[MAX],b[MAX];
11  int top,n;
12  double cross(point a,point b,point c){return (b-a)^(c-a);}//Triangle'Area*Area
13  double dis(point a,point b){return (a-b)*(a-b);}//Point Dis
14  bool cmp(point a,point b)
15  {
16      double tmp=cross(p[0],a,b);
17      if(tmp>eps||(fabs(tmp)<eps&&dis(p[0],a)-dis(p[0],b)>eps)) return 1;
18      return 0;
19  }
20  void graham()
21  {
22      int u=0;top=0;
23      for(int k=1;k<n;k++) if(p[u].y-p[k].y>eps||(fabs(p[u].y-p[k].y)<eps&&p[u].x-p[k].x>
        eps)) u=k;
24      swap(p[u],p[0]);
25      sort(p+1,p+n,cmp);
26      if(n>0) {b[0]=p[0];top++;}
27      if(n>1) {b[1]=p[1];top++;}
28      if(n<3) return ;
29      for(int i=2;i<n;i++)
30      {
31          while(top>1&&cross(b[top-2],b[top-1],p[i])<eps) top--;
32          b[top++]=p[i];
33      }
34  }
35  /*Zhou Chang*/
36  int main()
37  {
38      double sum=0; scanf("%d",&n);
39      for(int k=0;k<n;k++) scanf("%lf%lf",&p[k].x,&p[k].y);
40      graham();b[top]=b[0];
41      for(int k=1;k<=top;k++) sum+=sqrt(dis(b[k],b[k-1]));
42      printf("%.2f\n",sum);
43      return 0;
```

```
}
/* Area */
int main()
{
    double sum=0; scanf("%d",&n);
    for(int k=0;k<n;k++) scanf("%lf%lf",&p[k].x,&p[k].y);
    graham();
    for(int k=1;k<top-1;k++) sum+=fabs(cross(b[0],b[k],b[k+1]));//Sum of Triangle Area
    printf("%d\n",(int)(sum/100));
    return 0;
}
/*Farthest point Dis*/
double rotating()
{
    double ans=0; b[top]=b[0];
    for(int k=0,i=1;k<=top;k++)
    {
        while(fabs(cross(b[k],b[i+1],b[k+1]))-fabs(cross(b[k],b[i],b[k+1]))>eps) i=(i
    +1)%top;
        ans=max(ans,dis(b[i],b[k]));
    }
    return ans;
}
/*Minimum width*/
double rotating()
{
    double ans=0x3f3f3f3f; b[top]=b[0];
    for(int k=0,i=1;k<=top;k++)
    {
        while(fabs(cross(b[k],b[i+1],b[k+1]))-fabs(cross(b[k],b[i],b[k+1]))>eps) i=(i
    +1)%top;
        ans=min(ans,fabs(cross(b[k],b[k+1],b[i])/sqrt(dis(b[k],b[k+1]))));
    }
    return ans;
}
/* Max Area of Triangle*/
double rotating()
{
    double ans=0;
    for(int k=0;k<top;k++)
    {
        int i=(k+1)%top,j=(i+1)%top;
        while(i!=k&&j!=k)
        {
            ans=max(ans,fabs(cross(b[k],b[i],b[j])));
            while(fabs(cross(b[k],b[i+1],b[j]))-fabs(cross(b[k],b[i],b[j]))>eps) i=(i
    +1)%top;
            j=(j+1)%top;
        }
    }
    return ans;
}
/* The max/min Dis of two convex hull */
const double eps=1e-8;
const double INF=1e99;
struct point{}p[MAX],b1[MAX],b2[MAX];
int top1,top2;
double min(double a,double b){return a-b<-eps?a:b;}
double cross(point a,point b,point c){return (b-a)^(c-a);}
```

```
100 double multi(point a,point b,point c){return (b-a)*(c-a);}
101 double dis(point a,point b){return (a-b)*(a-b);}
102 double dist(point a,point b,point c)//Line Dis
103 {
104     point d;
105     double t=multi(b,a,c)/dis(b,c);
106     if(t>-eps&&t-1<eps) d=point(b.x+(c.x-b.x)*t,b.y+(c.y-b.y)*t);
107     else
108     {
109         if(dis(a,b)-dis(a,c)<-eps) d=b;
110         else d=c;
111     }
112     return dis(a,d);
113 }
114 double distence(point a,point b,point c,point d){return min(min(dist(a,c,d),dist(b,c,d)
    ),min(dist(c,a,b),dist(d,a,b)));}//Two Line
115 bool cmp(point a,point b)
116 {
117     double tmp=cross(p[0],a,b);
118     if(tmp>eps||(fabs(tmp)<eps&&dis(p[0],a)-dis(p[0],b)>eps)) return 1;
119     return 0;
120 }
121 void graham(point *b,int n,int &top)
122 {
123     int u=0;top=0;
124     for(int k=1;k<n;k++) if(p[u].y-p[k].y>eps||(fabs(p[u].y-p[k].y)<eps&&p[u].x-p[k].x>
    eps)) u=k;
125     swap(p[u],p[0]);
126     sort(p+1,p+n,cmp);
127     if(n>0) {b[0]=p[0];top++;}
128     if(n>1) {b[1]=p[1];top++;}
129     if(n<3) return ;
130     for(int i=2;i<n;i++)
131     {
132         while(top>1&&cross(b[top-2],b[top-1],p[i])<eps) top--;
133         b[top++]=p[i];
134     }
135 }
136 double rotating(point *a,int n,point *b,int m)
137 {
138     int i1=0,i2=0;
139     for(int k=0;k<n;k++) if(a[k].y-a[i1].y<-eps) i1=k;
140     for(int k=0;k<m;k++) if(b[k].y-b[i2].y>eps) i2=k;
141     a[n]=a[0];b[m]=b[0];
142     double tmp,ans=INF;
143     for(int k=0;k<n;k++)
144     {
145         while((tmp=cross(a[i1+1],b[i2+1],a[i1])-cross(a[i1+1],b[i2],a[i1]))>eps) i2=(i2
    +1)%m;
146         if(tmp<-eps) ans=min(ans,dist(b[i2],a[i1],a[i1+1]));
147         else ans=min(ans,distence(a[i1],a[i1+1],b[i2],b[i2+1]));
148         i1=(i1+1)%n;
149     }
150     return ans;
151 }
152 int main()
153 {
154     int n,m;
155     while(~scanf("%d%d",&n,&m)&&(n||m))
```

```
156     {
157         for(int k=0;k<n;k++) scanf("%lf%lf",&p[k].x,&p[k].y);
158         graham(b1,n,top1);
159         for(int k=0;k<m;k++) scanf("%lf%lf",&p[k].x,&p[k].y);
160         graham(b2,m,top2);
161         printf("%0.5f\n",sqrt(min(rotating(b1,top1,b2,top2),rotating(b2,top2,b1,top1)))
    );
162     }
163     return 0;
164 }
165 /* Minest rectangle contain convex hull*/
166 const double eps=1e-8;
167 const double INF=1e99;
168 struct point{}p[MAX],b[MAX];
169 int top,n;
170 double min(double a,double b){return a-b<-eps?a:b;}
171 double cross(point a,point b,point c){return (b-a)^(c-a);}
172 double multi(point a,point b,point c){return (b-a)*(c-a);}
173 double dis(point a,point b){return (a-b)*(a-b);}
174 bool cmp(point a,point b)
175 {
176     double tmp=cross(p[0],a,b);
177     if(tmp>eps||(fabs(tmp)<eps&&dis(p[0],a)-dis(p[0],b)>eps)) return 1;
178     return 0;
179 }
180 void graham()
181 {
182     int u=0;top=0;
183     for(int k=1;k<n;k++) if(p[u].y-p[k].y>eps||(fabs(p[u].y-p[k].y)<eps&&p[u].x-p[k].x>
    eps)) u=k;
184     swap(p[u],p[0]);
185     sort(p+1,p+n,cmp);
186     if(n>0) {b[0]=p[0];top++;}
187     if(n>1) {b[1]=p[1];top++;}
188     if(n<3) return ;
189     for(int i=2;i<n;i++)
190     {
191         while(top>1&&cross(b[top-2],b[top-1],p[i])<eps) top--;
192         b[top++]=p[i];
193     }
194 }
195 double rotating()
196 {
197     int r=1,u=1,l;
198     double ans=INF;
199     b[top]=b[0];
200     for(int k=0;k<top;k++)
201     {
202         while(cross(b[k],b[k+1],b[u+1])-cross(b[k],b[k+1],b[u])>-eps) u=(u+1)%top;
203         while(multi(b[k],b[k+1],b[r+1])-multi(b[k],b[k+1],b[r])>eps) r=(r+1)%top;
204         l=k==0?r:l;
205         while(multi(b[k],b[k+1],b[l+1])-multi(b[k],b[k+1],b[l])<eps) l=(l+1)%top;
206         ans=min(ans,fabs(cross(b[k],b[k+1],b[u]))*fabs(multi(b[k],b[k+1],b[r])-multi(b[
    k],b[k+1],b[l]))/dis(b[k],b[k+1]));
207     }
208     return ans;
209 }
210 int main()
211 {
```

```
    int t;scanf("%d",&t);
    for(int tc=1;tc<=t;tc++)
    {
        scanf("%d",&n);n*=4;
        for(int k=0;k<n;k++) scanf("%lf%lf",&p[k].x,&p[k].y);
        graham();
        printf("Case #%d:\n%d\n",tc,(int)(rotating()+0.5));
    }
    return 0;
}
```

4.3 Circle

```
/* Perimeter intersection*/
double cal(point c1,double r1,point c2,double r2)
{
    double d = sqrt(dis(c1,c2));
    if(r1+r2-d<-eps||fabs(r1-r2)-d>eps) return 0;
    double x = (d*d + r1*r1 - r2*r2)/(2*d);
    double t1 = acos(x / r1);
    double t2 = acos((d - x)/r2);
    return (t2*r2-t1*r1)*2;
}
/* Area intersection*/
double areaover(point c1,double r1,point c2,double r2)
{
    double d=sqrt(dis(c1,c2));
    if(r1+r2-d<-eps) return 0;
    if(fabs(r1-r2)-d>eps)
    {
        double r=min(r1,r2);
        return PI*r*r;
    }
    double x=(d*d+r1*r1-r2*r2)/(2*d);
    double t1=acos(x/r1);
    double t2=acos((d-x)/r2);
    return r1*r1*t1+r2*r2*t2-d*r1*sin(t1);
}
/ Triangle Outcenter */
point outcenter(point a,point b,point c)
{
    double a1=b.x-a.x,b1=b.y-a.y,c1=(a1*a1+b1*b1)/2;
    double a2=c.x-a.x,b2=c.y-a.y,c2=(a2*a2+b2*b2)/2;
    double d=a1*b2-a2*b1;
    return point(a.x+(c1*b2-c2*b1)/d,a.y+(a1*c2-a2*c1)/d);
}
```

5 Dynamic Programming

5.1 Math$_B it$

```
int L,R,a[20];long long dp[20][2];
long long dfs(int pos,int fa,int st,bool limit)
{
    if(pos==-1) return 1;
    if(!limit&&dp[pos][st]!=-1) return dp[pos][st];
    int up=limit?a[pos]:9;
    long long ans=0;
    for(int i=0;i<=up;++i)
    {
        if(i==4) continue;
        else if(fa==6&&i==2) continue;
        ans+=dfs(pos-1,i,i==6,limit&&i==a[pos]);
    }
    if(!limit) dp[pos][st]=ans;
    return ans;
}
long long get(long long x)
{
    int pos=0;
    while(x){a[pos++]=x%10;x/=10;}
    return dfs(pos-1,0,0,1);
}
int main()
{
    while(~scanf("%d%d",&L,&R))
    {
        if(L==0&&R==0) break;
        memset(dp,-1,sizeof(dp));
        printf("%lld\n",get(R)-get(L-1));
    }
    return 0;
}
```

5.2 Region

```
/* dp[i][j]=min(dp[i][j],dp[k][j-1]+add[k+1][i]) */
int a[MAX1],dp[MAX1][MAX2],add[MAX1][MAX1],s[MAX1][MAX2];
void init (int n,int m)
{
    for (int k=1;k<=n;k++) read(a[k]);
    for (int i=1;i<n;i++)
    for (int j=i+1;j<=n;j++)
        add[i][j]=add[i][j-1]+a[j]-a[(i+j)>>1];
    for (int k=1;k<=n;k++)
    {
        dp[k][1]=add[1][k];
        s[k][1]=1;
    }
}
int main ()
{
    int n,m;read(n);read(m);init(n,m);
    for (int i=2;i<=m;i++)
    {
```

```
20          s[n+1][i]=n;
21          for (int j=n;j>i;j--)
22          {
23              dp[j][i]=INF;
24              for (int k=s[j][i-1];k<=s[j+1][i];k++) if(dp[j][i]>dp[k][i-1]+add[k+1][j])
25              {
26                  dp[j][i]=dp[k][i-1]+add[k+1][j];
27                  s[j][i]=k;
28              }
29          }
30      }
31      printf ("%d\n",dp[n][m]);
32      return 0;
33  }
```

6 Others

6.1 Mod

```
int add(int x,int y){return (x+=y)>=MOD?x-MOD:x;}
int sub(int x,int y){return (x-=y)<0?x+MOD:x;}
int mul(int x,int y){return 1ll*x*y%MOD;}
```

6.2 Three$_{D}ivide$

```
for(int i=0;i<100;++i)//1e-6
{
    midl=l+(r-l)/3.0;midr=r-(r-l)/3.0;
    lr=getr(midl),rr=getr(midr);
    if(lr-rr>eps) l=midl;
    else r=midr;
}
```

6.3 Mo$_{A}lgorithm$

```
/* number of continue region sum of xor ==k*/
int n,m,K,a[MAX],b[MAX],cnt[MAX],block;
struct Q{int l,r,id;long long ans;}q[MAX];
bool cmp1(Q a,Q b){return a.l/block==b.l/block?a.r<b.r:a.l/block<b.l/block;}
bool cmp2(Q a,Q b){return a.id<b.id;}
int main()
{
    while(~scanf("%d%d%d",&n,&m,&K))
    {
        b[0]=0;memset(cnt,0,sizeof(cnt));a[0]=0;
        for(int i=1;i<=n;++i) {scanf("%d",&a[i]);b[i]=b[i-1]^a[i];}
        for(int i=0;i<m;++i) {scanf("%d%d",&q[i].l,&q[i].r);q[i].id=i;--q[i].l;}
        block=sqrt(n);
        sort(q,q+m,cmp1);
        int L=q[0].l,R=q[0].l-1;long long ans=0;
        for(int i=0;i<m;++i)
        {
            while(R<q[i].r) {++R;ans+=cnt[b[R]^K];++cnt[b[R]];}
            while(R>q[i].r) {--cnt[b[R]];ans-=cnt[b[R]^K];--R;}
            while(L>q[i].l) {--L;ans+=cnt[b[L]^K];++cnt[b[L]];}
            while(L<q[i].l) {--cnt[b[L]];ans-=cnt[b[L]^K];++L;}
            q[i].ans=ans;
        }
        sort(q,q+m,cmp2);
        for(int i=0;i<m;++i) printf("%lld\n",q[i].ans);
    }
    return 0;
}
/* Mo with update:the min number not appear in thr times(of the numbers appear) */
const int MAX=2e5+5;
int n,m,cntq,cntc,cntid,block,a[MAX],now[MAX],vis[MAX],ans[MAX];
map<int,int>id;
struct P{int l,r,t,ans;void is(int x1,int x2,int x3){l=x1;r=x2;t=x3;}}q[MAX];
struct Q{int pos,u,v,t;void is(int x1,int x2,int x3,int x4){pos=x1;u=x2;v=x3;t=x4;}}c[
    MAX];
inline bool cmp1(P& x1,P& x2)
{
```

```
    if(x1.l/block!=x2.l/block) return x1.l<x2.l;
    if(x1.r/block!=x2.r/block) return x1.r<x2.r;
    return x1.t<x2.t;
}
inline bool cmp2(P& x1,P& x2){return x1.t<x2.t;}
inline void add(int x,int d)
{
    if(vis[x]>0)--ans[vis[x]];
    vis[x]+=d;
    if(vis[x]>0)++ans[vis[x]];
}
inline void change(int &x,int l,int r,int val)
{
    if(c[x].v==0) return ;
    if(c[x].pos>=l&&c[x].pos<=r)
    {
        add(a[c[x].pos],-1);
        add(val,1);
    }
    a[c[x].pos]=val;
}
inline int sid(int x) {if(id[x]) return id[x]; return id[x]=++cntid;}
int main()
{
    int x,y,z;
    id.clear();cntq=cntid=0;memset(now,0,sizeof(now));memset(c,0,sizeof(c));
    scanf("%d%d",&n,&m);block=pow(n,2.0/3);
    for(int i=1;i<=n;++i) {scanf("%d",&a[i]);now[i]=a[i]=sid(a[i]);}
    for(int i=1;i<=m;++i)
    {
        scanf("%d%d%d",&x,&y,&z);
        if(x==1) q[cntq++].is(y,z,i);
        else c[i].is(y,now[y],z=sid(z),i),now[y]=z;
    }
    sort(q,q+cntq,cmp1);
    int l=1,r=0,t=0;
    for(int i=0;i<cntq;++i)
    {
        while(r<q[i].r) add(a[++r],1);
        while(l<q[i].l) add(a[l++],-1);
        while(r>q[i].r) add(a[r--],-1);
        while(l>q[i].l) add(a[--l],1);
        while(t<q[i].t) ++t,change(t,l,r,c[t].v);
        while(t>q[i].t) change(t,l,r,c[t].u),--t;
        int k=1;while(ans[k]) ++k;q[i].ans=k;
    }
    sort(q,q+cntq,cmp2);
    for(int i=0;i<cntq;++i) printf("%d\n",q[i].ans);
    return 0;
}
```

6.4 Simulated$_{Annealing}$

```
const int MAX=55;const double eps=1e-9;int n;
struct P{double x,y,z;}p[MAX];
double dist(P ta,P tb){return sqrt((ta.x-tb.x)*(ta.x-tb.x)+(ta.y-tb.y)*(ta.y-tb.y)+(ta.
    z-tb.z)*(ta.z-tb.z));}
double getans()
```

```
5  {
6      P s=p[0];
7      double t=100;//change
8      double dt=0.98;//change
9      double ans=1e9;
10     while(t>eps)
11     {
12         int k=0;
13         for(int i=0;i<n;++i) if(dist(s,p[i])>dist(s,p[k])) k=i;
14         double d=dist(s,p[k]);
15         ans=min(ans,d);
16         s.x+=(p[k].x-s.x)/d*t;
17         s.y+=(p[k].y-s.y)/d*t;
18         s.z+=(p[k].z-s.z)/d*t;
19         t*=dt;
20     }
21     return ans;
22 }
23 int main()
24 {
25     while(~scanf("%d",&n))
26     {
27         if(n==0) break;
28         for(int i=0;i<n;++i) scanf("%lf%lf%lf",&p[i].x,&p[i].y,&p[i].z);
29         printf("%.5f\n",getans());
30     }
31 }
```