Contents

积分表

Integrals of Rational Functions

$$\int \frac{1}{1+x^2}dx = \tan^{-1} x \tag{1}$$

$$\int \frac{1}{a^2+x^2}dx = \frac{1}{a}\tan^{-1}\frac{x}{a} \tag{2}$$

$$\int \frac{x}{a^2+x^2}dx = \frac{1}{2}\ln|a^2+x^2| \tag{3}$$

$$\int \frac{x^2}{a^2+x^2}dx = x - a\tan^{-1}\frac{x}{a} \tag{4}$$

$$\int \frac{x^3}{a^2+x^2}dx = \frac{1}{2}x^2 - \frac{1}{2}a^2\ln|a^2+x^2| \tag{5}$$

$$\int \frac{1}{ax^2+bx+c}dx = \frac{2}{\sqrt{4ac-b^2}}\tan^{-1}\frac{2ax+b}{\sqrt{4ac-b^2}} \tag{6}$$

$$\int \frac{1}{(x+a)(x+b)}dx = \frac{1}{b-a}\ln\frac{a+x}{b+x},\ a\neq b \tag{7}$$

$$\int \frac{x}{(x+a)^2}dx = \frac{a}{a+x} + \ln|a+x| \tag{8}$$

$$\int \frac{x}{ax^2+bx+c}dx = \frac{1}{2a}\ln|ax^2+bx+c| - \frac{b}{a\sqrt{4ac-b^2}}\tan^{-1}\frac{2ax+b}{\sqrt{4ac-b^2}} \tag{9}$$

Integrals with Roots

$$\int \frac{x}{\sqrt{x\pm a}}dx = \frac{2}{3}(x\mp 2a)\sqrt{x\pm a} \tag{10}$$

$$\int \sqrt{\frac{x}{a-x}}dx = -\sqrt{x(a-x)} - a\tan^{-1}\frac{\sqrt{x(a-x)}}{x-a} \tag{11}$$

$$\int \sqrt{\frac{x}{a+x}}dx = \sqrt{x(a+x)} - a\ln\left[\sqrt{x}+\sqrt{x+a}\right] \tag{12}$$

$$\int x\sqrt{ax+b}dx = \frac{2}{15a^2}(-2b^2+abx+3a^2x^2)\sqrt{ax+b} \tag{13}$$

$$\int \sqrt{x(ax+b)}dx = \frac{1}{4a^{3/2}}\left[(2ax+b)\sqrt{ax(ax+b)} - b^2\ln\left|a\sqrt{x}+\sqrt{a(ax+b)}\right|\right] \tag{14}$$

$$\int \sqrt{x^3(ax+b)}dx = \left[\frac{b}{12a} - \frac{b^2}{8a^2x} + \frac{x}{3}\right]\sqrt{x^3(ax+b)} + \frac{b^3}{8a^{5/2}}\ln\left|a\sqrt{x}+\sqrt{a(ax+b)}\right| \tag{15}$$

$$\int \sqrt{x^2\pm a^2}dx = \frac{1}{2}x\sqrt{x^2\pm a^2} \pm \frac{1}{2}a^2\ln\left|x+\sqrt{x^2\pm a^2}\right| \tag{16}$$

$$\int \sqrt{a^2-x^2}dx = \frac{1}{2}x\sqrt{a^2-x^2} + \frac{1}{2}a^2\tan^{-1}\frac{x}{\sqrt{a^2-x^2}} \tag{17}$$

$$\int x\sqrt{x^2\pm a^2}dx = \frac{1}{3}\left(x^2\pm a^2\right)^{3/2} \tag{18}$$

$$\int \frac{1}{\sqrt{x^2\pm a^2}}dx = \ln\left|x+\sqrt{x^2\pm a^2}\right| \tag{19}$$

$$\int \frac{1}{\sqrt{a^2-x^2}}dx = \sin^{-1}\frac{x}{a} \tag{20}$$

$$\int \frac{x}{\sqrt{x^2\pm a^2}}dx = \sqrt{x^2\pm a^2} \tag{21}$$

$$\int \frac{x}{\sqrt{a^2-x^2}}dx = -\sqrt{a^2-x^2} \tag{22}$$

$$\int \frac{x^2}{\sqrt{x^2\pm a^2}}dx = \frac{1}{2}x\sqrt{x^2\pm a^2} \mp \frac{1}{2}a^2\ln\left|x+\sqrt{x^2\pm a^2}\right| \tag{23}$$

$$\int \sqrt{ax^2+bx+c}dx = \frac{b+2ax}{4a}\sqrt{ax^2+bx+c} + \frac{4ac-b^2}{8a^{3/2}}\ln\left|2ax+b+2\sqrt{a(ax^2+bx^+c)}\right| \tag{24}$$

$$\int x\sqrt{ax^2+bx+c} = \frac{1}{48a^{5/2}}\left(2\sqrt{a}\sqrt{ax^2+bx+c} \times\left(-3b^2+2abx+8a(c+ax^2)\right) +3(b^3-4abc)\ln\left|b+2ax+2\sqrt{a}\sqrt{ax^2+bx+c}\right|\right) \tag{25}$$

$$\int \frac{1}{\sqrt{ax^2+bx+c}}dx = \frac{1}{\sqrt{a}}\ln\left|2ax+b+2\sqrt{a(ax^2+bx+c)}\right| \tag{26}$$

$$\int \frac{x}{\sqrt{ax^2+bx+c}}dx = \frac{1}{a}\sqrt{ax^2+bx+c} - \frac{b}{2a^{3/2}}\ln\left|2ax+b+2\sqrt{a(ax^2+bx+c)}\right| \tag{27}$$

$$\int \frac{dx}{(a^2+x^2)^{3/2}} = \frac{x}{a^2\sqrt{a^2+x^2}} \tag{28}$$

Integrals with Logarithms

$$\int \frac{\ln ax}{x} dx = \frac{1}{2}(\ln ax)^2 \tag{29}$$

$$\int \ln(ax+b)dx = \left(x+\frac{b}{a}\right)\ln(ax+b) - x, a \neq 0 \tag{30}$$

$$\int \ln(x^2+a^2)\,\mathrm{dx} = x\ln(x^2+a^2) + 2a\tan^{-1}\frac{x}{a} - 2x \tag{31}$$

$$\int \ln(x^2-a^2)\,\mathrm{dx} = x\ln(x^2-a^2) + a\ln\frac{x+a}{x-a} - 2x \tag{32}$$

$$\int \ln\left(ax^2+bx+c\right)dx = \frac{1}{a}\sqrt{4ac-b^2}\tan^{-1}\frac{2ax+b}{\sqrt{4ac-b^2}} - 2x + \left(\frac{b}{2a}+x\right)\ln\left(ax^2+bx+c\right) \tag{33}$$

$$\int x\ln(ax+b)dx = \frac{bx}{2a} - \frac{1}{4}x^2 + \frac{1}{2}\left(x^2-\frac{b^2}{a^2}\right)\ln(ax+b) \tag{34}$$

$$\int x\ln\left(a^2-b^2x^2\right)dx = -\frac{1}{2}x^2 + \frac{1}{2}\left(x^2-\frac{a^2}{b^2}\right)\ln\left(a^2-b^2x^2\right) \tag{35}$$

Integrals with Exponentials

$$\int x^n e^{ax}\,\mathrm{dx} = \frac{x^n e^{ax}}{a} - \frac{n}{a}\int x^{n-1}e^{ax}\,\mathrm{dx} \tag{36}$$

$$\int xe^{-ax^2}\,\mathrm{dx} = -\frac{1}{2a}e^{-ax^2} \tag{37}$$

Integrals with Trigonometric Functions

$$\int \sin^3 ax dx = -\frac{3\cos ax}{4a} + \frac{\cos 3ax}{12a} \tag{38}$$

$$\int \cos^2 ax dx = \frac{x}{2} + \frac{\sin 2ax}{4a} \tag{39}$$

$$\int \cos^3 ax dx = \frac{3\sin ax}{4a} + \frac{\sin 3ax}{12a} \tag{40}$$

$$\int \cos ax \sin bx dx = \frac{\cos[(a-b)x]}{2(a-b)} - \frac{\cos[(a+b)x]}{2(a+b)}, a \neq b \tag{41}$$

$$\int \sin^2 ax \cos bx dx = -\frac{\sin[(2a-b)x]}{4(2a-b)} + \frac{\sin bx}{2b} - \frac{\sin[(2a+b)x]}{4(2a+b)} \tag{42}$$

$$\int \sin^2 x \cos x dx = \frac{1}{3}\sin^3 x \tag{43}$$

$$\int \cos^2 ax \sin bx dx = \frac{\cos[(2a-b)x]}{4(2a-b)} - \frac{\cos bx}{2b} - \frac{\cos[(2a+b)x]}{4(2a+b)} \tag{44}$$

$$\int \cos^2 ax \sin ax dx = -\frac{1}{3a}\cos^3 ax \tag{45}$$

$$\int \sin^2 ax \cos^2 bx dx = \frac{x}{4} - \frac{\sin 2ax}{8a} - \frac{\sin[2(a-b)x]}{16(a-b)} + \frac{\sin 2bx}{8b} - \frac{\sin[2(a+b)x]}{16(a+b)} \tag{46}$$

$$\int \sin^2 ax \cos^2 ax dx = \frac{x}{8} - \frac{\sin 4ax}{32a} \tag{47}$$

$$\int \tan ax dx = -\frac{1}{a}\ln\cos ax \tag{48}$$

$$\int \tan^2 ax dx = -x + \frac{1}{a}\tan ax \tag{49}$$

$$\int \tan^3 ax dx = \frac{1}{a}\ln\cos ax + \frac{1}{2a}\sec^2 ax \tag{50}$$

$$\int \sec x dx = \ln|\sec x + \tan x| = 2\tanh^{-1}\left(\tan\frac{x}{2}\right) \tag{51}$$

$$\int \sec^2 ax dx = \frac{1}{a}\tan ax \tag{52}$$

$$\int \sec^3 x\,\mathrm{dx} = \frac{1}{2}\sec x\tan x + \frac{1}{2}\ln|\sec x + \tan x| \tag{53}$$

$$\int \sec x \tan x dx = \sec x \tag{54}$$

$$\int \sec^2 x \tan x dx = \frac{1}{2}\sec^2 x \tag{55}$$

$$\int \sec^n x \tan x dx = \frac{1}{n}\sec^n x, n \neq 0 \tag{56}$$

$$\int \csc x dx = \ln\left|\tan\frac{x}{2}\right| = \ln|\csc x - \cot x| + C \tag{57}$$

$$\int \csc^2 ax dx = -\frac{1}{a}\cot ax \tag{58}$$

$$\int \csc^3 x dx = -\frac{1}{2}\cot x\csc x + \frac{1}{2}\ln|\csc x - \cot x| \tag{59}$$

$$\int \csc^n x \cot x dx = -\frac{1}{n}\csc^n x, n \neq 0 \tag{60}$$

$$\int \sec x \csc x dx = \ln|\tan x| \tag{61}$$

Products of Trigonometric Functions and Monomials

$$\int x\cos x dx = \cos x + x\sin x \tag{62}$$

$$\int x\cos ax dx = \frac{1}{a^2}\cos ax + \frac{x}{a}\sin ax \tag{63}$$

$$\int x^2\cos x dx = 2x\cos x + \left(x^2-2\right)\sin x \tag{64}$$

$$\int x^2\cos ax dx = \frac{2x\cos ax}{a^2} + \frac{a^2x^2-2}{a^3}\sin ax \tag{65}$$

$$\int x\sin x dx = -x\cos x + \sin x \tag{66}$$

$$\int x\sin ax dx = -\frac{x\cos ax}{a} + \frac{\sin ax}{a^2} \tag{67}$$

$$\int x^2\sin x dx = \left(2-x^2\right)\cos x + 2x\sin x \tag{68}$$

$$\int x^2\sin ax dx = \frac{2-a^2x^2}{a^3}\cos ax + \frac{2x\sin ax}{a^2} \tag{69}$$

Products of Trigonometric Functions and Exponentials

$$\int e^x\sin x dx = \frac{1}{2}e^x(\sin x - \cos x) \tag{70}$$

$$\int e^{bx}\sin ax dx = \frac{1}{a^2+b^2}e^{bx}(b\sin ax - a\cos ax) \tag{71}$$

$$\int e^x\cos x dx = \frac{1}{2}e^x(\sin x + \cos x) \tag{72}$$

$$\int e^{bx}\cos ax dx = \frac{1}{a^2+b^2}e^{bx}(a\sin ax + b\cos ax) \tag{73}$$

$$\int xe^x\sin x dx = \frac{1}{2}e^x(\cos x - x\cos x + x\sin x) \tag{74}$$

$$\int xe^x\cos x dx = \frac{1}{2}e^x(x\cos x - \sin x + x\sin x) \tag{75}$$

Dynamic Hull

```cpp
bool __slp__x__;
template<typename T>
struct HULL{
  struct node{
    T slp,x,y;
    inline node(T _slp=0,T _x=0,T _y=0){slp=_slp;x=_x;y=_y;}
    inline bool operator<(const node&a)const{return __slp__x__?slp>a.slp:x<a.x;}
    inline T operator-(const node&a)const{return(y-a.y)/(x-a.x);}
  };
  set<node>Q;
  inline void add(T x,T y){
    __slp__x__=0;
    node t(0,x,y);
    typename set<node>::iterator it=Q.lower_bound(node(0,x,0));
    if(it!=Q.end()){
      if((it->x==x&&it->y>=y)||(it->x!=x&&it->slp<=*it-t))return;
      if(it->x==x)Q.erase(it);
    }it=Q.insert(t).c0;
    typename set<node>::iterator it3=it;it3--;
    while(it!=Q.begin()){
      typename set<node>::iterator it2=it3;
      if(it2!=Q.begin()&&t-*it2>=*it2-*--it3)Q.erase(it2);
      else break;
    }it3=it;it3++;
    while(it3!=Q.end()){
      typename set<node>::iterator it2=it3;
      if(++it3!=Q.end()&&*it2-*it3>=*it2-t)Q.erase(it2);
      else break;
    }if(it==Q.begin())const_cast<T&>(it->slp)=1e9;
    else{
      typename set<node>::iterator it2=it;it2--;
      const_cast<T&>(it->slp)=t-*it2;
    }typename set<node>::iterator it2=it;it2++;
    if(it2!=Q.end())const_cast<T&>(it2->slp)=t-*it2;
  }inline pair<T,T>get(T a,T b){
    //min(ax+by)
    if(Q.empty())return mp(0,0);
    __slp__x__=1;
    typename set<node>::iterator it=Q.lower_bound(node(-a/b,0,0));
    if(it!=Q.begin())it--;
    return mp(it->x,it->y);
  }
};
```

lyndon

```cpp
namespace lyndon{
  vector<int> work(char *s,int n){
    int i=1;vector<int> res;res.clear();
    while(i<=n){
      int j=i;
      int k=i+1;
      while(k<=n&&s[j]<=s[k]){
        if(s[j]<s[k])j=i;
        else j++;
        k++;
      }
      while(i<=j){
        res.push_back(i);
        i+=k-j;
      }
    }
    return res;
  }
};
```

SAM

```cpp
namespace sam{
  const int N=1010000;int go[N][26],len[N],fail[N],tot,last;
  void initsam(){rep(i,1,tot){len[i]=fail[i]=0;rep(j,0,25)go[i][j]=0;}tot=last=1;}
  // Rev(S) 的后缀自动机建出来就是 S 的后缀树
  // 对于结点 x,fail[x] 到 x 的边是 right[x]-len[fail[x]]..right[x]-len[x]+1 ( 倒着的
  ↪ )
  // 这个板子没有求 right, 需要的话自己写个拓扑排序求
  void add(int c,int pos){
    int p=last;int np=++tot;len[np]=len[p]+1;last=np;
    for(;p&&(!go[p][c]);p=fail[p])go[p][c]=np;
    if(!p){
      fail[np]=1;//lct::newson(1,np,pos);
      return;
    }
    int gt=go[p][c];
    if(len[p]+1==len[gt]){
      fail[np]=gt;//lct::newson(gt,np,pos);
      return;
    }
    int nt=++tot;len[nt]=len[p]+1;fail[nt]=fail[gt];fail[gt]=nt;
    //lct::cut(fail[nt],gt,nt);
    fail[np]=nt;
    //lct::newson(nt,np,pos);
```

```
  rep(i,0,25)go[nt][i]=go[gt][i];for(;p&&go[p][c]==gt;p=fail[p])go[p][c]=nt;
 }
};
```

exkmp

```
void ex_kmp(char s[], int next[], int n) {
 //s[1..next[i]]=s[i..i+next[i]-1]
  int i, a = 0, l = 0, p = 0;
  for (i = 2, next[1] = n; i <= n; ++i) {
    l = max(min(next[i - a + 1], p - i + 1), 0);
    for (; i + l <= n && s[1 + l] == s[i + l]; ++l);
    next[i] = l;
    if (i + l - 1 > p) a = i, p = i + l - 1;
  }
}
```

Manacher

```
namespace manacher{
 const int N=110000;
 char ch[N<<1],s[N];
 int f[N<<1],id,mx,n,len;
 // 以 i 为中心对应 f[i*2], 以 (i,i+1) 为中心对应 f[i*2+1]
 // f[i]-1 为回文串长度
 void init(char *s){
   n=strlen(s); ch[0]='$'; ch[1]='#';
   for (int i=1;i<=n;i++){
     ch[i*2]=s[i-1]; ch[i*2+1]='#';
   }
   id=0; mx=0; ch[n*2+2]='#';
   for (int i=0;i<=2*n+10;i++) f[i]=0;
   for (int i=1;i<=2*n+2;i++){
     if (i>mx) f[i]=1; else f[i]=min(f[id*2-i],mx-i);
     while (ch[i-f[i]]==ch[i+f[i]]) f[i]++;
     if (i+f[i]>mx){mx=i+f[i]; id=i;}
   }
 }
}
```

Maximum Express

```
// 找字典序最大的循环表示 要求 s 下标从 0 开始，并扩展到 2 倍
int lex_find(char s[], int n, bool rev) {
  int a = 0, b = 1, l;
  while (a < n && b < n) {
    for (l = 0; l < n; ++l)
      if (s[a + l] != s[b + l]) break;
    if (l < n) {
      if (s[a + l] > s[b + l]) b = b + l + 1;
      else a = a + l + 1;
      if (a == b) ++b;
    } else {
      if (a > b) swap(a, b);
      if (rev) return n - (b - a) + a;
      else return a;
    }
  }
  return min(a, b);
}
```

Suffix Array

```
namespace SA{
 const int N=110000;
 int n,m,p,x[N],y[N],c[N],sa[N],rank[N],height[N];
 char ch[N];
 void make(){
  for (int i=1;i<=m;i++) c[i]=0;
  for (int i=1;i<=n;i++) c[x[i]]++;
  for (int i=1;i<=m;i++) c[i]+=c[i-1];
  for (int i=1;i<=n;i++){
   sa[c[x[i]]]=i; c[x[i]]--;
  }
  int k=1;
  while (k<n){
   p=0;
   for (int i=n-k+1;i<=n;i++) y[++p]=i;
   for (int i=1;i<=n;i++) if (sa[i]>k) y[++p]=sa[i]-k;
   for (int i=1;i<=m;i++) c[i]=0;
   for (int i=1;i<=n;i++) c[x[i]]++;
   for (int i=1;i<=m;i++) c[i]+=c[i-1];
   for (int i=n;i;i--) sa[c[x[y[i]]]--]=y[i];
   for (int i=1;i<=n;i++) y[i]=x[i];
   p=1; x[sa[1]]=1;
   for (int i=2;i<=n;i++){
    if (y[sa[i]]!=y[sa[i-1]]||y[sa[i]+k]!=y[sa[i-1]+k]) p++; x[sa[i]]=p;
   }
   if (p==n) break; m=p; k<<=1;
  }
  for (int i=1;i<=n;i++) rank[sa[i]]=i;
  k=0;
  for (int i=1;i<=n;i++){
```

```
      if (rank[i]==1) continue;
      if (k) k--;
      while (ch[i+k]==ch[sa[rank[i]-1]+k]) k++;
      height[rank[i]]=k;
    }
  }
  void init(char *s){
    n=strlen(s);
    for (int i=0;i<=n+10;i++) height[i]=0;
    for (int i=1;i<=n;i++) x[i]=s[i-1],ch[i]=x[i]; x[n+1]=-1; ch[n+1]=-1;
    m=200; make();
  }
  // init 的时候把字符串传进去就可以了
  // sa 和 height 同定义
}
```

Ukk

```
namespace UKK{
const int maxN=1010000,inf=1e9;
struct tree{
    int l,r,go[26],father,link,d,id,e;
}t[maxN<<1];
struct state{
    int where,rem,d;
}now;
int len,S[maxN],N,en,Q[maxN],Ql,Qr;
long long ans;
// ans 表示所有边的长度和 , 统计子串个数时要减去无穷边的数量 ( 叶子节点是无穷边 )
void newnode(){
  len++; t[len].l=t[len].r=t[len].father=t[len].link=t[len].d=t[len].id=t[len].e=0;
  memset(t[len].go,0x00,sizeof t[len].go);
}
state follow(state now,int way){
    if (now.rem==0){
        if (t[now.where].go[way]==0) return (state){0,0,0};
        else {
            int k1=t[now.where].go[way]; return (state){k1,t[k1].r-t[k1].l-1,now.d+1};
        }
    } else if (S[t[now.where].r-now.rem]==way) return (state){now.where,now.rem-1,now.d+1};
    else return (state){0,0,0};
}
state go(int now,int l,int r){
    while (l<r){
        int k1=t[now].go[S[l]];
        if (t[k1].r-t[k1].l>=r-l) return (state){k1,t[k1].r-t[k1].l-(r-l),t[now].d+(r-l)};
        now=k1; l+=t[k1].r-t[k1].l;
    }
    return (state){now,0,t[now].d};
}
void change(int l,int r,int pre,int where){
    t[where].father=pre; t[where].l=l; t[where].r=r;
    t[where].d=t[pre].d+r-l; t[pre].e+=(t[pre].go[S[l]]==0);
    t[pre].go[S[l]]=where;
    if (r==inf) t[where].id=inf-t[where].d,t[pre].id=max(t[pre].id,t[where].id);
}
int splite(state now){
    if (now.rem==0) return now.where;
    if (now.rem==t[now.where].r-t[now.where].l) return t[now.where].father;
    newnode();
    change(t[now.where].l,t[now.where].r-now.rem,t[now.where].father,len);
    int k1=S[t[len].r]; t[len].go[k1]=now.where; t[now.where].father=len;
    t[now.where].l=t[len].r; t[len].id=t[now.where].id; t[len].e=1;
    return len;
}
int getlink(int k){
    if (t[k].link) return t[k].link;
    t[k].link=splite(go(getlink(t[k].father),t[k].l+(t[k].father==1),t[k].r));
    return t[k].link;
}
void insert(int k){ // push back
    S[++N]=k;
    while (1){
        state ne=follow(now,k);
        if (ne.where){
            now=ne; return;
        }
        int mid=splite(now);
        newnode(); int leaf=len;
        change(N,inf,mid,leaf); int newnod=getlink(mid);
        now.where=newnod; now.rem=0; now.d=t[newnod].d;
        Q[++Qr]=leaf; ans+=t[leaf].r-t[leaf].l; en++;
        if (mid==1) return;
    }
}
long long getsubstring(){
    return ans-1ll*(inf-N-1)*en;
}
```

```cpp
void del(){ // pop front
    Ql++;
    int where=Q[Ql];
    ans-=t[where].r-t[where].l; if (t[where].r==inf) en--;
    if (where==now.where){
        now=go(getlink(t[where].father),t[where].l+(t[where].father==1),t[where].r-now.rem);
        int prel=t[where].l;
        t[where].l=Qr+t[t[where].father].d+1; t[where].d=t[t[where].father].d+t[where].r-t[where].l;
        t[where].id=Qr+1; if (where==now.where) now.rem+=prel-t[where].l;
        ans+=t[where].r-t[where].l; Q[++Qr]=where; en+=(t[where].r==inf);
        return;
    }
    t[t[where].father].go[S[t[where].l]]=0; t[t[where].father].e--;
    if (t[t[where].father].e==1&&t[where].father!=1){
        int newson=0,r=t[where].father,rr=t[r].father;
        for (int i=0;i<26;i++) if (t[r].go[i]) newson=t[t[where].father].go[i];
        int pre=t[newson].r-t[newson].l;
        int newl=t[newson].r-t[newson].l+t[r].r-t[r].l;
        if (t[newson].e)
            change(t[newson].id+t[rr].d,t[newson].id+t[rr].d+newl,rr,newson);
        else change(t[newson].id+t[rr].d,inf,rr,newson);
        if (now.where==r) now.where=newson,now.rem+=pre;
    }
}
void init(){
  len=0; newnode(); now=(state){1,0,0}; t[1].link=1; en=0; Ql=Qr=0; N=0; ans=0;
}
}
```

回文树

```cpp
const int M=26;int fail[N];
int go[N][M],len[N],diff[N],anc[N],lst;
int n;char str[N];int p;int s[N];int f[N],g[N];
void addChar(int c,int ww){
  int x=lst;while(s[ww]!=s[ww-len[x]-1])x=fail[x];// ww 是位置 ， 下标从 1 开始
  if(!go[x][c]){
    len[p]=len[x]+2;int k=fail[x];while(s[ww]!=s[ww-len[k]-1])k=fail[k];fail[p]=go[k][c];
    go[x][c]=p;diff[p]=len[p]-len[fail[p]];
    if(diff[p]==diff[fail[p]])anc[p]=anc[fail[p]]; // anc[x] 表示祖先中 ， 第一个和 x 不在一个等差数列里的回文串
    else anc[p]=fail[p];p++;
  }
  lst=go[x][c]; // 求长度 ， 直接倒着 for 一遍即可
}
void init(){
  rep(i,1,p){anc[i]=diff[i]=len[i]=fail[i]=0;rep(j,0,M-1)go[i][j]=0;}
  p=2;len[0]=0;len[1]=-1;fail[0]=1; //node 1: 所有奇数长度的串的祖先
  fail[1]=0;f[0]=1;lst=1;
}
void work(){
  s[0]=-1;init(); //s[0] 要设成字符集之外的数
  rep(i,1,n){
    addChar(s[i],i);
    for(int x=lst;x;x=anc[x]){
      g[x]=f[i-(len[anc[x]]+diff[x])]; //g[x] 记录包含 x 的等差数列链的信息 (x 一定是链底 )
      if(anc[x]!=fail[x])g[x]=(g[x]+g[fail[x]])%P;
      if(i%2==0)f[i]=(f[i]+g[x])%P;
      /*  当新加入一个字符 ， 扩展整个链时 .g[x] 被扩展到时 ,fail[x] 上一次被扩展到一定是在 i-d 时
        假设这一段等差数列里的长度是 l[1]..l[m],l[m] 是 g[x] 代表的点 ， 则 g[fail[x]]=sum(j=1..m-1)f[i-d-l[j]]
        而 g[x]=sum(j=1..m)f[i-l[j]]=f[i-l[1]]+sum(j=2..m)f[i-l[j-1]-d]=f[i-l[1]]+g[fail[x]]=f[i-(len[anc[x]]+diff[x])]+g[fail[x]]
        可以认为 ,g[x] 是在 g[fail[x]] 的基础上 ， 添加了 i-l[1] 这个左边界
        注意 ， 如果是维护其他的信息 ， 注意把 g[x] 以前的贡献去除掉 */
    }
  }
}
```

KM

```cpp
namespace KM{
typedef long long i64;
const int maxN = 401;
const int oo = 0x3f3f3f3f;
int vx[maxN],vy[maxN],lx[maxN],ly[maxN],slack[maxN];
int w[maxN][maxN]; // 以上为权值类型
int pre[maxN],left[maxN],right[maxN],NL,NR,N;
void match(int& u) {
  for(;u; std::swap(u, right[pre[u]]))
    left[u] = pre[u];
}
void bfs(int u) {
  static int q[maxN], front, rear;
  front = 0; vx[q[rear = 1] = u] = true;
  while(true) {
    while(front < rear) {
```

```cpp
      int u = q[++front];
      for(int v = 1;v <= N; ++v) {
        int tmp;
        if(vy[v] || (tmp = lx[u] + ly[v] - w[u][v]) > slack[v])
          continue;
        pre[v] = u;
        if(!tmp) {
          if(!left[v]) return match(v);
          vy[v] = vx[q[++rear] = left[v]] = true;
        } else slack[v] = tmp;
      }
    }

    int a = oo;
    for(int i = 1;i <= N; ++i)
      if(!vy[i] && a > slack[i]) a = slack[u = i];
    for(int i = 1;i <= N; ++i) {
      if(vx[i]) lx[i] -= a;
      if(vy[i]) ly[i] += a;
      else slack[i] -= a;
    }
    if(!left[u]) return match(u);
    vy[u] = vx[q[++rear] = left[u]] = true;
  }
}
void exec() {
  for(int i = 1;i <= N; ++i) {
    for(int j = 1;j <= N; ++j) {
      slack[j] = oo;
      vx[j] = vy[j] = false;
    }
    bfs(i);
  }
}
i64 work(int nl,int nr){// NL , NR 为左右点数 , 返回最大权匹配的权值和
  NL=nl;NR=nr;
  N=std::max(NL,NR);
  for(int u = 1;u <= N; ++u)
    for(int v = 1;v <= N; ++v){
      lx[u] = std::max(lx[u], w[u][v]);
    }

  exec();

  i64 ans = 0;
  for(int i = 1;i <= N; ++i)
    ans += lx[i] + ly[i];
  return ans;
}
void output(){ // 输出左边点与右边哪个点匹配 , 没有匹配输出 0
  for(int i = 1;i <= NL; ++i)
    printf("%d ",(w[i][right[i]] ? right[i] : 0));
  printf("\n");
}
}
```

带花树

```cpp
namespace KM{
typedef long long i64;
const int maxN = 401;
const int oo = 0x3f3f3f3f;
int vx[maxN],vy[maxN],lx[maxN],ly[maxN],slack[maxN];
int w[maxN][maxN]; // 以上为权值类型
int pre[maxN],left[maxN],right[maxN],NL,NR,N;
void match(int& u) {
  for(;u; std::swap(u, right[pre[u]]))
    left[u] = pre[u];
}
void bfs(int u) {
  static int q[maxN], front, rear;
  front = 0; vx[q[rear = 1] = u] = true;
  while(true) {
    while(front < rear) {
      int u = q[++front];
      for(int v = 1;v <= N; ++v) {
        int tmp;
        if(vy[v] || (tmp = lx[u] + ly[v] - w[u][v]) > slack[v])
          continue;
        pre[v] = u;
        if(!tmp) {
          if(!left[v]) return match(v);
          vy[v] = vx[q[++rear] = left[v]] = true;
        } else slack[v] = tmp;
      }
    }

    int a = oo;
    for(int i = 1;i <= N; ++i)
      if(!vy[i] && a > slack[i]) a = slack[u = i];
    for(int i = 1;i <= N; ++i) {
```

```
      if(vx[i]) lx[i] -= a;
      if(vy[i]) ly[i] += a;
      else slack[i] -= a;
    }
    if(!left[u]) return match(u);
    vy[u] = vx[q[++rear] = left[u]] = true;
  }
}
void exec() {
  for(int i = 1;i <= N; ++i) {
    for(int j = 1;j <= N; ++j) {
      slack[j] = oo;
      vx[j] = vy[j] = false;
    }
    bfs(i);
  }
}
i64 work(int nl,int nr){// NL , NR 为左右点数 , 返回最大权匹配的权值和
  NL=nl;NR=nr;
  N=std::max(NL,NR);
  for(int u = 1;u <= N; ++u)
    for(int v = 1;v <= N; ++v){
      lx[u] = std::max(lx[u], w[u][v]);
    }

  exec();

  i64 ans = 0;
  for(int i = 1;i <= N; ++i)
    ans += lx[i] + ly[i];
  return ans;
}
void output(){ // 输出左边点与右边哪个点匹配 , 没有匹配输出 0
  for(int i = 1;i <= NL; ++i)
    printf("%d ",(w[i][right[i]] ? right[i] : 0));
  printf("\n");
}
}
```

2-sat

```
namespace Twosat{
  const int M=4000010,N=1000010;
  struct bian{
    int next,point;
  }b[M];
  int n,len,p[N<<1],dfs[N<<1],low[N<<1],where[N<<1],now,sign;
  int s[N<<1],head,in[N<<1],ans[N],ou[N<<1],sign2;
  void add(int k1,int k2){
    b[++len]=(bian){p[k1],k2}; p[k1]=len;
  }
  // n 表示限制个数 ,i 为取 ,i+n 为不取
  // 连边一定要对称
  void init(int _n){
    n=_n;
    for (int i=0;i<=n*2+1;i++){
      p[i]=where[i]=dfs[i]=low[i]=s[i]=in[i]=ou[i]=0;
    }
    len=now=sign=head=sign2=0;
  }
  void tarjan(int k1){
    s[++head]=k1; in[k1]=1; dfs[k1]=++sign; low[k1]=sign;
    for (int i=p[k1];i;i=b[i].next){
      int j=b[i].point;
      if (dfs[j]==0){
        tarjan(j); low[k1]=min(low[k1],low[j]);
      } else if (in[j]) low[k1]=min(low[k1],dfs[j]);
    }
    if (dfs[k1]==low[k1]){
      now++;
      while (1){
        where[s[head]]=now; in[s[head]]=0;
        ou[s[head]]=++sign2; head--;
        if (s[head+1]==k1) break;
      }
    }
  }
  // ans[i]=0 表示取 i, 否则表示取 i+n
  int solve(){
    for (int i=1;i<=n*2;i++) if (dfs[i]==0) tarjan(i);
    for (int i=1;i<=n;i++) if (where[i]==where[i+n]) return 0;
    for (int i=1;i<=n;i++) if (ou[i]<ou[i+n]) ans[i]=0; else ans[i]=1;
    return 1;
  }
}
```

Cactus

```
namespace Cactus{
  const int NN=51000,M=101000;
  struct bian{
    int next,point;
```

```
}b[M<<1];
int p[NN],len,n,m,pd[M<<1],father[NN],d[NN];
int N,u[M],v[M],A[NN+M];
vector<int>go[NN+M];
// A 每一个环最上方的节点
// 将仙人掌建成圆方树 , 编号 >n 的节点为环 ,d 为深度
// 把边 add 进去之后调用 buildtree, 树存在 go 中
// 如果不一定是仙人掌则要事先判断 , 不然复杂度会爆炸
void init(int _n){
  n=_n; len=-1;
  memset(p,0xff,sizeof p);
  memset(pd,0x00,sizeof pd);
  memset(d,0x00,sizeof d);
  memset(father,0x00,sizeof father);
  for (int i=1;i<=n+m;i++) go[i].clear();
}
void ade(int k1,int k2){
  b[++len]=(bian){p[k1],k2}; p[k1]=len;
}
void Add(int k1,int k2){
  ade(k1,k2); ade(k2,k1);
}
void add(int k1,int k2){
  m++; u[m]=k1; v[m]=k2; Add(k1,k2);
}
void dfs(int k1,int k2){
  father[k1]=k2; d[k1]=d[k2]+1;
  for (int i=p[k1];i!=-1;i=b[i].next){
    int j=b[i].point;
    if (d[j]==0){
      pd[(i>>1)+1]=1; dfs(j,k1);
    }
  }
}
int compare(int k1,int k2){
  return d[k1]<d[k2];
}
void buildtree(){
  dfs(1,0); N=n;
  for (int i=1;i<=m;i++)
    if (pd[i]==0){
      int k1=u[i],k2=v[i]; N++;
      if (d[k1]<d[k2]) swap(k1,k2); A[N]=k2;
      while (k1!=k2){
        // 不是仙人掌的话在这儿判一下每一条边只被覆盖一次
        int pre=father[k1]; father[k1]=N; k1=pre;
      }
      go[k2].push_back(N);
    }
  for (int i=2;i<=n;i++){
    go[father[i]].push_back(i);
  }
}
}
```

点双

```
namespace CC{
// 先调用 init() 点双数 num
// 点双树在 go 中 , 每一个点双向割点连边 , pd>1 为割点 , pd=0 为孤立点
const int N=110000;
struct bian{
  int next,point;
}b[N<<1];
vector<int>E[N<<1],V[N<<1],go[N<<1];
int num,s[N<<1],head,pd[N],dfs[N],low[N],sign,loc[N];
int p[N],len,n;
void ade(int k1,int k2){b[++len]=(bian){p[k1],k2}; p[k1]=len;}
void add(int k1,int k2){ade(k1,k2); ade(k2,k1);}
void solve(int k1){
  dfs[k1]=++sign; low[k1]=dfs[k1];
  for (int i=p[k1];i!=-1;i=b[i].next){
    int j=b[i].point;
    if (dfs[j]==0){
      s[++head]=i; solve(j); low[k1]=min(low[k1],low[j]);
      if (low[j]==dfs[k1]){
        num++;
        while (1){
          E[num].push_back(s[head]); head--;
          if (s[head+1]==i) break;
        }
      }
    } else if (dfs[j]<dfs[k1])
      s[++head]=i,low[k1]=min(low[k1],dfs[j]);
  }
}
void work(){
  for (int i=1;i<=n;i++) if (dfs[i]==0) solve(i);
  sign=0;
  for (int i=1;i<=num;i++){
    sign++;
```

```
    for (int j=0;j<E[i].size();j++){
      int k1=b[E[i][j]].point;
      if (pd[k1]!=sign){
        pd[k1]=sign; V[i].push_back(k1);
      }
    }
  }
  for (int i=1;i<=num;i++)
    for (int j=0;j<V[i].size();j++) pd[V[i][j]]++;
  for (int i=1;i<=num;i++)
    for (int j=0;j<V[i].size();j++)
      if (pd[V[i][j]]>1){
        go[V[i][j]].push_back(i+n);
        go[i+n].push_back(V[i][j]);
      } else loc[V[i][j]]=i;
}
void init(int _n){
  n=_n; sign=head=0; len=-1; num=0;
  for (int i=1;i<=n;i++) pd[i]=dfs[i]=low[i]=loc[i]=0;
  for (int i=1;i<=n*2;i++) go[i].clear(),V[i].clear(),E[i].clear();
}
}
```

zkw

```
namespace Flow{
  const int M=100010,N=1010,inf=1e9;
  struct bian{
    int next,point,f,w;
  }b[M];
  int totpoint,p[N],len,n,m,D[N],pd[N],sign,flow,cost,bo[N];
  // D 为顶标 ， 对残量网络满足最短路那个不等式
  void ade(int k1,int k2,int k3,int k4){
    b[++len]=(bian){p[k1],k2,k3,k4}; p[k1]=len;
  }
  void add(int k1,int k2,int k3,int k4){
    n=max(n,k1); n=max(n,k2);
    ade(k1,k2,k3,k4); ade(k2,k1,0,-k4);
  }
  void init(int _totpoint){
    memset(p,0xff,sizeof p); len=-1; flow=0; cost=0;
    memset(D,0x00,sizeof D);
    totpoint=_totpoint; n=totpoint;
  }
  int dfs(int k1,int k2){
    pd[k1]=sign;
    if (k1==totpoint||k2==0) return k2;
    for (int i=p[k1];i!=-1;i=b[i].next){
      int j=b[i].point;
      if (b[i].f&&D[j]==D[k1]+b[i].w&&pd[j]!=sign){
        int k=dfs(j,min(k2,b[i].f));
        if (k==0) continue;
        b[i].f-=k; b[i^1].f+=k;
        cost+=k*b[i].w;
        return k;
      }
    }
    return 0;
  }
  int newD(){
    if (pd[totpoint]==sign) return 1;
    int w=inf;
    for (int now=0;now<=n;now++)
      if (pd[now]==sign)
        for (int i=p[now];i!=-1;i=b[i].next){
          int j=b[i].point;
          if (b[i].f&&pd[j]!=sign) w=min(w,D[now]-D[j]+b[i].w);
        }
    if (w==inf) return 0;
    for (int i=0;i<=n;i++) if (pd[i]==sign) D[i]-=w;
    return 1;
  }
  void get(){
    do{
      sign++; flow+=dfs(0,inf);
    }while(newD());
  }
}
```

最小树形图

```
namespace ZL{
  // a 尽量开大 ， 之后的边都塞在这个里面
  const int N=100010,M=100010,inf=1e9;
  struct bian{
      int u,v,w,use,id;
  }b[M],a[2000100];
  int n,m,ans,pre[N],id[N],vis[N],root,In[N],h[N],len,way[M];
  // 从 root 出发能到达每一个点的最小支撑树
  // 先调用 init 然后把边 add 进去 ， 需要方案就 getway,way[i] 为 1 表示使用
  void init(int _n,int _root){
    n=_n; m=0; b[0].w=1e9; root=_root;
```

```
}
void add(int u,int v,int w){
  m++; b[m]=(bian){u,v,w,0,m}; a[m]=b[m];
}
int work(){
  len=m;
    for (;;){
        for (int i=1;i<=n;i++){pre[i]=0; In[i]=inf; id[i]=0; vis[i]=0; h[i]=0;}
        for (int i=1;i<=m;i++) if (b[i].u!=b[i].v&&b[i].w<In[b[i].v]){
            pre[b[i].v]=b[i].u; In[b[i].v]=b[i].w; h[b[i].v]=b[i].id;
        }
        for (int i=1;i<=n;i++) if (pre[i]==0&&i!=root) return 0;
        int cnt=0; In[root]=0;
        for (int i=1;i<=n;i++){
            if (i!=root) a[h[i]].use++; int now=i; ans+=In[i];
            while (vis[now]==0&&now!=root){ vis[now]=i; now=pre[now]; }
            if (now!=root&&vis[now]==i){
                cnt++; int kk=now;
                while (1){
                    id[now]=cnt; now=pre[now];
                    if (now==kk) break;
                }
            }
        }
        if (cnt==0) return 1; for (int i=1;i<=n;i++) if (id[i]==0) id[i]=++cnt;
        // 缩环 , 每一条接入的边都会茶包原来接入的那条边 , 所以要调整边权
        // 新加的边是 u, 茶包的边是 v
        for (int i=1;i<=m;i++){
            int k1=In[b[i].v]; int k2=b[i].v; b[i].u=id[b[i].u];
↪ b[i].v=id[b[i].v];
            if (b[i].u!=b[i].v){
                b[i].w-=k1; a[++len].u=b[i].id; a[len].v=h[k2]; b[i].id=len;
            }
        }
        n=cnt; root=id[root];
    }
    return 1;
}
void getway(){
  for (int i=1;i<=m;i++) way[i]=0;
  for (int i=len;i>m;i--){ a[a[i].u].use+=a[i].use; a[a[i].v].use-=a[i].use; }
  for (int i=1;i<=m;i++) way[i]=a[i].use;
}
}
```

Diameter Tree

```
//Floyd First
for(i=-1;++i!=n;) {
  for(j=-1;++j!=n;r[j][0]=dis[i][r[j][1]]);
  qsort(r,n,8,cmp);
  for(j=i;++j!=n;)
    if(map[i][j]!=0x3F3F3F3F) {
      for(d=x=0;++x!=n;)
        if(dis[j][r[x][1]]>dis[j][r[d][1]])
          ans=min(ans,map[i][j]+dis[i][r[x][1]]+dis[j][r[d][1]]),d=x;
    if(!d) ans=min(ans,min(dis[j][r[0][1]],r[0][0])<<1);
  }
}
```

Dominator Tree

```
namespace dominator{
  // DAG 的 dominator tree 可以直接 LCA 做
  // 最开始先 init() 传入点数 , 通过 add 加边 , 出发点编号为 1 可能需要重标号
  // dominator tree 的结构存在 go 中 , 可能存在点无法到达即不在树中
  // go 中的下标是根据 dfs 序重标号过的
  // semi i 的祖先 x, 不经过 i 到 x 之间树上的点能到达 i 的最高祖先
  const int N=110000,M=1010000;
  struct bian{
      int next,point;
  }b[M];
  int dfs[N],x[N],p[N],len,pre[N];
  int idom[N],best[N],semi[N],f[N];
  vector<int>go[N];
  void ade(int k1,int k2){
      b[++len]=(bian){p[k1],k2}; p[k1]=len;
  }
  void add(int k1,int k2){
      ade(k1,k2); ade(k2,k1);
  }
  // 先通过一次 dfs 给所有点标号 , 如果已经给出了标号这一步可以省略
  void solve(int k){
      dfs[k]=++len; x[len]=k;
      for (int i=p[k];i!=-1;i=b[i].next){
          int j=b[i].point; if (i&1) continue;
          if (dfs[j]==0) {solve(j); pre[dfs[j]]=dfs[k];}
      }
  }
  int get(int k){
      if (k==f[k]) return k;
      int k1=get(f[k]);
```

```
   if (semi[best[k]]>semi[best[f[k]]]) best[k]=best[f[k]];
   f[k]=k1; return f[k];
 }
 void tarjan(){
   for (int now=len;now>=2;now--){
     int k1=x[now];
     for (int i=p[k1];i!=-1;i=b[i].next){
       if ((i&1)==0) continue;
       int j=dfs[b[i].point];
       if (j==0) continue; get(j);
       if (semi[best[j]]<semi[now]) semi[now]=semi[best[j]];
     }
     go[semi[now]].push_back(now);
     int k2=pre[now]; f[now]=pre[now];
     for (int i=0;i<go[k2].size();i++){
       int j=go[k2][i];
       get(j);
       if (semi[best[j]]<k2) idom[j]=best[j]; else idom[j]=k2;
     }
     go[k2].clear();
   }
   for (int i=2;i<=len;i++){
     if (semi[i]!=idom[i]) idom[i]=idom[idom[i]];
     go[idom[i]].push_back(i);
   }
 }
 void init(int n){
  len=-1;
  for (int i=1;i<=n;i++){
   p[i]=-1,f[i]=best[i]=semi[i]=i,go[i].clear();
   idom[i]=0,pre[i]=x[i]=dfs[i]=0;
  }
 }
 void getdominator(){
  len=0; solve(1); tarjan();
 }
}
```

SS-algorithm

```
const int N=55;
int n,m;
struct perm{
  int p[N];
  inline perm(int ise=0){rep(i,1,n)p[i]=i*ise;}
  inline perm inv(){perm res;rep(i,1,n)res.p[p[i]]=i;return res;}
};
vector<perm> T[N];perm R[N][N];
inline int sz(int x){int ans=0;rep(i,1,n)ans+=R[x][i].p[1]>0;return ans;}
inline perm operator *(const perm &a,const perm &b){
  perm c;rep(i,1,n)c.p[i]=a.p[b.p[i]];return c;
}
//-------------------permutation-----------------------------------
bool check(perm x,int k){
  //check if x in <S>
  return (!k)||(R[k][x.p[k]].p[1]&&check(R[k][x.p[k]]*x,k-1));
}
void dfs(perm x,int k);
void insert(perm x,int k){//insert(x,n)
  if(check(x,k))return;
  T[k].push_back(x);
  rep(i,1,n)if(R[k][i].p[1])dfs(x*R[k][i].inv(),k);
}
void dfs(perm x,int k){
  if(R[k][x.p[k]].p[1])insert(R[k][x.p[k]]*x,k-1);
  else{
    R[k][x.p[k]]=x.inv();
    rep(i,0,T[k].size()-1)dfs(T[k][i]*x,k);
  }
}
void init(){
  rep(i,1,n)rep(j,1,n)R[i][j]=perm(i==j);rep(i,1,n)T[i].clear();
}
```

洲阁筛

```
namespace Sieve{
 const int N=100000;//sqrt(N)
 const int S=100000;
 int inv[55];
 int vf(int x){return inv[2];}//V(p)
 int vg(int x,int c){return inv[c+1];}//V(p^c) (c>1)
 int Val(int p,int c){if(c==1)return vf(p);else return vg(p,c);}//V(p^c) (c>=1)
 bool notp[N+10];int pr[N+10],prtot,w[N+10],m,pos[N+10],n,pre[N+10],small[N+10],f[N+10],g[N+10];
 int fd(int x){//find the largest prime that is no more than x
  int l=1;int r=prtot;int ret=0;
  while(l<r){int mid=(l+r)>>1;if(pr[mid]<=x)ret=mid,l=mid+1;else r=mid;}
  if(pr[l]<=x)ret=l;return ret;
 }
 int preSV(int p){return p*1ll*inv[2]%P;}//sum(x=1..p)V(pr[x])
 int sumF(int l,int r){return (r-l+1);}//sum(x=1..p)F(x)
```

```
int sumV(int l,int r){if(l>r)return 0;return (preSV(fd(r))+P-preSV(fd(l-1)))%P;}//sum(x=l..r&x is prime)V(x)
int sumV2(int l,int r){if(l>r)return 0;return (preSV(r)+P-preSV(l-1))%P;}//sum(x=l..r)V(pr[x])
int sumV3(int l,int r){if(l>r)return 0;return (r-l+1)%P;}//sum(x=l..r)F(pr[x])
int vfg(int x){return 1;}//F(x)
int getPos(int x){if(x<=S)return pos[x];else return m+1-pos[n/x];}
int getVal(int x,int t){return (g[x]+P-sumV3(pre[x]+1,min(t-1,small[x])))%P;}
void Main(int _n){
  rep(i,1,50)inv[i]=Pow(i,P-2);n=_n;
  for(int i=2;i<=N;++i){
    if(!notp[i])pr[++prtot]=i;
    for(int j=1;j<=prtot&&pr[j]*1ll*i<=N;++j){
      notp[i*pr[j]]=1;if(i%pr[j]==0)break;
    }
  }
  for(int i=1;i<=n;i=n/(n/i)+1)w[++m]=n/i;
  sort(w+1,w+1+m);rep(i,1,m)if(w[i]<=S)pos[w[i]]=i;
  f[getPos(n)]=1;int up=1;int ans=0;
  rep(i,1,m){small[i]=small[i-1];while(small[i]<prtot&&pr[small[i]+1]<=w[i])++small[i];}
  rep(i,1,prtot){
    int nup=up;
    rep(j,up,m){
      if(pr[i]>w[j]){
        nup=max(nup,j+1);continue;
      }
      if(pr[i]*pr[i]>w[j]){
        nup=max(nup,j+1);int res=f[j];res=res*1ll*sumV(pr[i],w[j])%P;ans=(ans+res)%P;continue;
      }
      for(int v=w[j]/pr[i],c=1;v;v/=pr[i],c++){
        int y=getPos(v);f[y]=(f[y]+f[j]*1ll*Val(pr[i],c))%P;
        if(pr[i]*pr[i]>w[y]){
          int s=f[j]*1ll*Val(pr[i],c)%P;s=s*1ll*sumV2(i+1,small[y])%P;ans=(ans+s)%P;
        }
      }
    }
    up=nup;
  }
  //G must meet G(ab)=G(a)G(b)
  rep(i,1,m)g[i]=sumF(1,w[i]);up=1;
  rep(i,1,prtot){
    int nup=up;
    per(j,m,up){
      if(pr[i]>w[j]){
        nup=max(nup,j+1);g[j]=1;pre[j]=i;continue;
      }
      g[j]=(g[j]+P-(vfg(pr[i])*1ll*getVal(getPos(w[j]/pr[i]),i)%P))%P;
      if(pr[i]*pr[i]>w[j]){nup=max(nup,j+1);pre[j]=i;continue;}
      pre[j]=i;
    }
    up=nup;
  }
  rep(i,1,m)g[i]=getVal(i,prtot+1);
  rep(i,1,m)ans=(ans+f[i]*1ll*(1+(g[i]+P-1)*1ll*inv[2]%P))%P;//need modify
  printf("%d\n",ans);
}
};
```

fft

```
#define upmo(a,b) (((a)=((a)+(b))%mo)<0?(a)+=mo:(a))
const db pi=3.14159265358979323846264338327L;const int FFT_MAXN=262144;int mo=2;
struct cp{
  db a,b;
  cp operator +(const cp&y)const{return (cp){a+y.a,b+y.b};}
  cp operator -(const cp&y)const{return (cp){a-y.a,b-y.b};}
  cp operator *(const cp&y)const{return (cp){a*y.a-b*y.b,a*y.b+b*y.a};}
  cp operator !()const{return cp{a,-b};};
}nw[FFT_MAXN+1];
int bitrev[FFT_MAXN];
void dft(cp*a,int n,int flag=1){
  int d=0;while((1<<d)*n!=FFT_MAXN)d++;
  rep(i,0,n-1)if(i<(bitrev[i]>>d))swap(a[i],a[bitrev[i]>>d]);
  for(int l=2;l<=n;l<<=1){
    int del=FFT_MAXN/l*flag;
    for(int i=0;i<n;i+=l){
      cp *le=a+i;cp *ri=a+i+(l>>1);
      cp *w=flag==1?nw:nw+FFT_MAXN;
      rep(k,0,(l>>1)-1){
        cp ne=*ri**w;*ri=*le-ne,*le=*le+ne;le++,ri++,w+=del;
      }
    }
  }
  if(flag!=1)rep(i,0,n-1)a[i].a/=n,a[i].b/=n;
}
void fft_init(){
  int L=0;while((1<<L)!=FFT_MAXN)L++;
  bitrev[0]=0;rep(i,1,FFT_MAXN-1)bitrev[i]=bitrev[i>>1]>>1|((i&1)<<(L-1));
```

```
  rep(i,0,FFT_MAXN)nw[i]=(cp){(db)cosl(2*pi/FFT_MAXN*i),(db)sinl(2*pi/FFT_MAXN*i)};
}
void convoP(int *a,int n,int *b,int m,int *c){ // 任意模数 fft, 需要提前设定 mo
  rep(i,0,n+m)c[i]=0;
  static cp f[FFT_MAXN],g[FFT_MAXN],t[FFT_MAXN];int N=2;while(N<=n+m)N<<=1;
  rep(i,0,N-1){
    int aa=i<=n?a[i]:0;int bb=i<=m?b[i]:0;
    upmo(aa,0);upmo(bb,0);
    f[i]=(cp){db(aa>>15),db(aa&32767)};
    g[i]=(cp){db(bb>>15),db(bb&32767)};
  }
  dft(f,N);dft(g,N);
  rep(i,0,N-1){int j=i?N-i:0;t[i]=((f[i]+!f[j])*(!g[j]-g[i])+(!f[j]-f[i])*(g[i]+!g[j]))*(cp){0,0.25};}
  dft(t,N,-1);
  rep(i,0,n+m)upmo(c[i],(ll(t[i].a+0.5))%mo<<15);
  rep(i,0,N-1){int j=i?N-i:0;t[i]=(!f[j]-f[i])*(!g[j]-g[i])*(cp){-0.25,0}+(cp){0,0.25}*(f[i]+!f[j])*(g[i]+!g[j]);}
  dft(t,N,-1);
  rep(i,0,n+m)upmo(c[i],ll(t[i].a+0.5)+(ll(t[i].b+0.5)%mo<<30));
}
void convoF(int *a,int n,int *b,int m,int *c,int P){ // 快速的 fft
  static cp f[FFT_MAXN>>1],g[FFT_MAXN>>1],t[FFT_MAXN>>1];
  int N=2; while (N<=n+m) N<<=1;
  rep(i,0,N-1){
    if (i&1){
      f[i>>1].b=(i<=n)?a[i]:0.0;g[i>>1].b=(i<=m)?b[i]:0.0;
    } else {
      f[i>>1].a=(i<=n)?a[i]:0.0;g[i>>1].a=(i<=m)?b[i]:0.0;
    }
  }
  dft(f,N>>1); dft(g,N>>1);int del=FFT_MAXN/(N>>1);
  cp qua=(cp){0,0.25},one=(cp){1,0},four=(cp){4,0},*w=nw;
  rep(i,0,(N>>1)-1){
    int j=i?(N>>1)-i:0;
    t[i]=(four*!(f[j]*g[j])-(!f[j]-f[i])*(!g[j]-g[i])*(one+*w))*qua;
    w+=del;
  }
  dft(t,N>>1,-1);
  rep(i,0,n+m) c[i]=((long long)(((i&1)?t[i>>1].a:t[i>>1].b)+0.5))%P;
}
```

ntt

```
const int P=998244353;
const int G=3;const int N=(1<<22)+5;
int rev[N],w[2][N];
inline void init(int n){
    rep(i,0,n-1){
        int x=0;int y=i;for(int k=1;k<n;k<<=1,y>>=1)(x<<=1)|=(y&1);rev[i]=x;
    }
    w[0][0]=w[1][0]=1;int cha=Pow(G,(P-1)/n);int cha2=Pow(cha,P-2);
    rep(i,1,n-1){
        w[0][i]=w[0][i-1]*1ll*cha%P;
        w[1][i]=w[1][i-1]*1ll*cha2%P;
    }
}
inline void NTT(int *A,int N,bool ms){
    for(int i=0;i<N;i++)if(i<rev[i]){
        int tmp=A[i];A[i]=A[rev[i]];A[rev[i]]=tmp;
    }
    for(int i=1;i<N;i<<=1){
        for(int j=0;j<N;j+=(i<<1)){
            for(int k=0,l=0;k<i;k++,l+=N/(i<<1)){
                int x,y;y=A[j+k];x=A[j+k+i]*1ll*w[ms][l]%P;
                A[j+k]=(x+y)%P;A[j+k+i]=(y-x+P)%P;
            }
        }
    }
    if(ms){
        int v=Pow(N,P-2);rep(i,0,N-1)A[i]=A[i]*1ll*v%P;
    }
}
```

BM

```
namespace BM{
  const int mo=1e9+7,L=31000; const long long N=5ll*mo*mo;
  int x[L],y[L],len,prelen,prep,A[L],n,z[L],prew;
  // 依次加入 A[i], 找到长度为 len 的递推式, 其中 sum A[j-len+i]*x[i]=0
  // 时间复杂度 O(n^2), 插入直接 addin(), 输出 x 数组即可
  // 求行列式可以随机两个向量乘成数列 , 然后利用这个把特征多项式求出来
  int check(int n){
    long long w=0;
    for (int i=0;i<=len;i++){
      w=(w+1ll*A[n-len+i]*x[i]); if (w>N) w-=N;
    }
    return w%mo;
  }
  int quick(int k1,int k2){
    int k3=1;
    while (k2){
```

```
      if (k2&1) k3=1ll*k3*k1%mo; k2>>=1; k1=1ll*k1*k1%mo;
    }
    return k3;
  }
  void addin(int k1){
    A[++n]=k1; int num=check(n); if (num==0) return;
    int last=prep-prelen,now=n-len,kk=1ll*prew*num%mo;
    if (now<=last){
      for (int i=last-now;i<=prelen+last-now;i++){
        x[i]=(x[i]-1ll*y[i-last+now]*kk)%mo; if (x[i]<0) x[i]+=mo;
      }
      return;
    }
    for (int i=0;i<=len;i++) z[i]=x[i];
    int shi=now-last;
    for (int i=len;i>=0;i--) x[i+shi]=x[i];
    for (int i=0;i<shi;i++) x[i]=0;
    for (int i=0;i<=prelen;i++){
      x[i]=(x[i]-1ll*y[i]*kk)%mo; if (x[i]<0) x[i]+=mo;
    }
    prelen=len; prep=n; prew=quick(num,mo-2); for (int i=0;i<=len;i++) y[i]=z[i];
    len+=shi;
  }
  void init(){
    memset(x,0x00,sizeof x); memset(y,0x00,sizeof y);
    memset(z,0x00,sizeof z); memset(A,0x00,sizeof A);
    prelen=0; y[0]=1; prep=0; len=0; x[0]=1; n=0; prew=0;
  }
};
```

Pollard Rho

```
namespace Pollard_Rho {
typedef long long ll;
inline ll gcd(ll a, ll b) {ll c; while (b) c=a%b, a=b, b=c; return a;}
inline ll mulmod(ll x, ll y, const ll z) {return (x*y-(ll)(((long
  ↪ double)x*y+0.5)/(long double)z)*z+z)%z;}
inline ll powmod(ll a, ll b, const ll mo) {
  ll s = 1;
  for (; b; b>>=1, a = mulmod(a, a, mo)) if(b&1) s = mulmod(s, a, mo);
  return s;
}
bool isPrime(ll p) { // Miller-Rabin
  const int lena = 10, a[lena] = {2,3,5,7,11,13,17,19,23,29};
  if (p == 2) return true;
  if (p == 1 || !(p&1)) return false;
  ll D = p - 1;while (!(D&1)) D >>= 1;
  for (int i = 0; i < lena && a[i] < p; i++) {
    ll d = D, t = powmod(a[i], d, p); if (t == 1) continue;
    for (; d != p - 1 && t != p - 1; d <<= 1) t = mulmod(t, t, p);
    if (d == p - 1) return false;
  }
  return true;
}
void reportFactor(ll n){ // 得到一个素因子
  ans=min(ans,n);
}
ll ran(){return rand();} // 随机数
void getFactor(ll n) { // Pollard-Rho
  if (n == 1) return;
  if (isPrime(n)) { reportFactor(n); return; }
  while (true) {
    ll c = ran() % n, i = 1, x = ran() % n, y = x, k = 2;
    do {
      ll d = gcd(n + y - x, n);
      if(d != 1 && d != n) { getFactor(d); getFactor(n / d); return; }
      if (++i == k) y = x, k <<= 1;
      x = (mulmod(x, x, n) + c) % n;
    } while (y != x);
  }
}
}
```

Simplex

```
namespace Simplex{
  // where,w,way 至少要开两倍 默认有变量 >=0 的限制
  double A[30][30];
  const double eps=1e-10;
  int n,m,where[70],M,flag,ifun;
  double ans,w[70],way[70];
  void init(int _n){
    memset(A,0x00,sizeof A); memset(where,0x00,sizeof where);
    memset(w,0x00,sizeof w); memset(way,0x00,sizeof way);
    n=m=M=flag=ifun=ans=0; n=_n;
  }
  void turn(int e,int l){
    swap(where[e],where[l+n]);
    for (int i=0;i<=M;i++)
      if (i!=l){
        double t=A[i][e]/A[l][e];
        for (int j=0;j<=n;j++)
```

```
        if (j!=e) A[i][j]-=t*A[l][j]; else A[i][e]=-t;
      }
    double pre=A[l][e]; A[l][e]=1;
    for (int i=0;i<=n;i++) A[l][i]/=pre;
  }
  double solve(){
    while (1){
      int e=0,l=0;
      for (int i=1;i<=n;i++) if (A[0][i]>eps) {
        if (e==0||where[i]<where[e]) e=i;
      }
      if (e==0){return -A[0][0];}
      for (int i=1;i<=m;i++)
        if (A[i][e]>eps){
          if (l==0||A[i][0]*A[l][e]<A[i][e]*A[l][0]-
  ↪ eps||(A[i][0]*A[l][e]<A[i][e]*A[l][0]+eps&&where[i+n]<where[l+n]))
  ↪ l=i;
        }
      if (l==0){ifun=1; return 0;}
      turn(e,l);
    }
  }
  int getans(){ // 0 表示无解 ,1 表示无穷大 ,2 表示存在最大值
    n++; int l=1;
    for (int i=1;i<=m;i++) A[i][n]=-1; A[0][n]=-1;
    for (int i=1;i<=n+M;i++) where[i]=i;
    for (int i=2;i<=m;i++) if (A[i][0]<A[l][0]) swap(l,i);
    if (A[l][0]<0) turn(n,l);
    if (solve()<-eps) return 0;
    m++; for (int i=0;i<=n;i++) swap(A[0][i],A[m][i]),A[m][i]=-A[m][i];
    ans=solve(); if (ifun) return 1;
    for (int i=1;i<=n-1;i++) w[i]=0;
    for (int i=1;i<=m;i++) w[i+n]=A[i][0];
    for (int i=1;i<=n-1;i++)
      for (int j=1;j<=n+m;j++) if (where[j]==i) way[i]=w[j];
    return 2;
  }
  void setcondition(double *x,double lim){ // x 为系数 ,lim 为小于等于多少
    m++; for (int i=1;i<=n;i++) A[m][i]=x[i]; A[m][0]=lim;
  }
  void setmaximal(double *x){ // x 为系数 , 要最大化多少 , 要在限制加完后在加
    for (int i=1;i<=n;i++) A[m+1][i]=x[i]; M=m+1;
  }
};
```

Int Simplex

```
namespace simplex{ // 默认有变量 >=0 的限制
typedef int db;
const int N=1000+5,M=10000+5,inf=1e9;
db a[M][N],b[M];
int idn[N],idm[M],nxt[N],n,m;
void init(int _n){ // nxt 数组不需要初始化
  n=_n;
  memset(a,0,sizeof(a)); memset(b,0,sizeof(b));
  memset(idn,0,sizeof(idn)); memset(idm,0,sizeof(idm));
}
void pivot(int x,int y){
  swap(idm[x],idn[y]);
    db k=a[x][y];b[x]/=k;a[x][y]=1/k;
    rep(j,1,n)a[x][j]/=k;  int t=n+1;
  for(int i=1;i<=n;i++)  if(a[x][i]){nxt[t]=i;t=i;nxt[t]=-1;}
  rep(i,0,m)if(i!=x){
    db k=a[i][y];  if(!k)continue;
    b[i]-=k*b[x],a[i][y]=0;
    for(int j=nxt[n+1];j!=-1;j=nxt[j])a[i][j]-=a[x][j]*k;
  }
}
void simplex(){
  idn[0]=inf;
  while(1){
    int y=0;
    rep(j,1,n)if(a[0][j]>0&&idn[j]<idn[y])y=j;
    if(!y)break;int x=0;
    rep(i,1,m)if(a[i][y]>0)
      if(!x) x=i;else{
        int t=b[i]/a[i][y]-b[x]/a[x][y];
        if(t<0||(t==0&&idm[i]<idm[x]))x=i;
      }
    if(!x){puts("Unbounded");  exit(0);}
    pivot(x,y);
  }
}
void init_solution(){
  rep(j,1,n)idn[j]=j;  rep(i,1,m)idm[i]=n+i;
  idm[0]=inf;idn[0]=inf;
  // 寻找初始解 , 如果全为 0 是一个合法的解那么以下过程不需要进行
  while(1){
    int x=0;rep(i,1,m)if(b[i]<0&&idm[i]<idm[x])x=i;
    if(!x)break;  int y=0;
    rep(j,1,n)if(a[x][j]<0&&idn[j]<idn[y])y=j;
```

```
    if(!y){puts("Infeasible");  exit(0);}  pivot(x,y);
  }
}
void output(){ // 输出方案
  rep(j,1,n){
    bool f=1;
    rep(i,1,m)if(idm[i]==j){printf("%d ",b[i]);f=1;break;}
    if(!f)printf("0 ");
  }
  puts("");
}
void setcondition(db *x,db lim){ // x 为系数 , lim 为小于等于多少
  m++; for (int i=1;i<=n;i++) a[m][i]=x[i]; b[m]=lim;
}
void setmaximal(db *x){  // x 为系数 , 要最大化多少 , 可以在限制加完前加
  for (int i=1;i<=n;i++) a[0][i]=x[i];
}
db solve(){
  init_solution();  simplex(); return -b[0];
}
}
```

Geometry2D

```
#define mp make_pair
#define fi first
#define se second
#define pb push_back
typedef double db;
const db eps=1e-6;
const db pi=acos(-1);
int sign(db k){
    if (k>eps) return 1; else if (k<-eps) return -1; return 0;
}
int cmp(db k1,db k2){return sign(k1-k2);}
int inmid(db k1,db k2,db k3){return sign(k1-k3)*sign(k2-k3)<=0;}// k3 在 [k1,k2] 内
struct point{
    db x,y;
    point operator + (const point &k1) const{return (point){k1.x+x,k1.y+y};}
    point operator - (const point &k1) const{return (point){x-k1.x,y-k1.y};}
    point operator * (db k1) const{return (point){x*k1,y*k1};}
    point operator / (db k1) const{return (point){x/k1,y/k1};}
    int operator == (const point &k1) const{return cmp(x,k1.x)==0&&cmp(y,k1.y)==0;}
    // 逆时针旋转
    point turn(db k1){return (point){x*cos(k1)-y*sin(k1),x*sin(k1)+y*cos(k1)};}
    point turn90(){return (point){-y,x};}
    bool operator < (const point k1) const{
        int a=cmp(x,k1.x);
        if (a==-1) return 1; else if (a==1) return 0; else return cmp(y,k1.y)==-1;
    }
    db abs(){return sqrt(x*x+y*y);}
    db abs2(){return x*x+y*y;}
    db dis(point k1){return ((*this)-k1).abs();}
    point unit(){db w=abs(); return (point){x/w,y/w};}
    void scan(){double k1,k2; scanf("%lf%lf",&k1,&k2); x=k1; y=k2;}
    void print(){printf("%.11lf %.11lf\n",x,y);}
    db getw(){return atan2(y,x);}
    point getdel(){if (sign(x)==-1||(sign(x)==0&&sign(y)==-1)) return (*this)*(-1); else return (*this);}
  int getP() const{return sign(y)==1||(sign(y)==0&&sign(x)==-1);}
};
int inmid(point k1,point k2,point k3){return inmid(k1.x,k2.x,k3.x)&&inmid(k1.y,k2.y,k3.y);}
db cross(point k1,point k2){return k1.x*k2.y-k1.y*k2.x;}
db dot(point k1,point k2){return k1.x*k2.x+k1.y*k2.y;}
db rad(point k1,point k2){return atan2(cross(k1,k2),dot(k1,k2));}
// -pi -> pi
int compareangle (point k1,point k2){
    return k1.getP()<k2.getP()||(k1.getP()==k2.getP()&&sign(cross(k1,k2))>0);
}
point proj(point k1,point k2,point q){ // q 到直线 k1,k2 的投影
    point k=k2-k1; return k1+k*(dot(q-k1,k)/k.abs2());
}
point reflect(point k1,point k2,point q){return proj(k1,k2,q)*2-q;}
int clockwise(point k1,point k2,point k3){// k1 k2 k3 逆时针 1 顺时针 -1 否则 0
    return sign(cross(k2-k1,k3-k1));
}
int checkLL(point k1,point k2,point k3,point k4){// 求直线 (L) 线段 (S)k1,k2 和 k3,k4 的交点
    return cmp(cross(k3-k1,k4-k1),cross(k3-k2,k4-k2))!=0;
}
point getLL(point k1,point k2,point k3,point k4){
    db w1=cross(k1-k3,k4-k3),w2=cross(k4-k3,k2-k3); return (k1*w2+k2*w1)/(w1+w2);
}
int intersect(db l1,db r1,db l2,db r2){
    if (l1>r1) swap(l1,r1); if (l2>r2) swap(l2,r2); return cmp(r1,l2)!=-1&&cmp(r2,l1)!=-1;
}
int checkSS(point k1,point k2,point k3,point k4){
    return intersect(k1.x,k2.x,k3.x,k4.x)&&intersect(k1.y,k2.y,k3.y,k4.y)&&
    sign(cross(k3-k1,k4-k1))*sign(cross(k3-k2,k4-k2))<=0&&
```

```
    sign(cross(k1-k3,k2-k3))*sign(cross(k1-k4,k2-k4))<=0;
}
db disSP(point k1,point k2,point q){
    point k3=proj(k1,k2,q);
    if (inmid(k1,k2,k3)) return q.dis(k3); else return min(q.dis(k1),q.dis(k2));
}
db disSS(point k1,point k2,point k3,point k4){
    if (checkSS(k1,k2,k3,k4)) return 0;
    else return min(min(disSP(k1,k2,k3),disSP(k1,k2,k4)),min(disSP(k3,k4,k1),disSP(k3,k4,k2)));
}
int onS(point k1,point k2,point q){return inmid(k1,k2,q)&&sign(cross(k1-q,k2-k1))==0;}
struct circle{
    point o; db r;
    void scan(){o.scan(); scanf("%lf",&r);}
    int inside(point k){return cmp(r,o.dis(k));}
};
struct line{
    // p[0]->p[1]
    point p[2];
    line(point k1,point k2){p[0]=k1; p[1]=k2;}
    point& operator [] (int k){return p[k];}
    int include(point k){return sign(cross(p[1]-p[0],k-p[0]))>0;}
    point dir(){return p[1]-p[0];}
    line push(){ // 向外 ( 左手边 ) 平移 eps
        const db eps = 1e-6;
        point delta=(p[1]-p[0]).turn90().unit()*eps;
        return {p[0]-delta,p[1]-delta};
    }
};
point getLL(line k1,line k2){return getLL(k1[0],k1[1],k2[0],k2[1]);}
int parallel(line k1,line k2){return sign(cross(k1.dir(),k2.dir()))==0;}
int sameDir(line k1,line k2){return parallel(k1,k2)&&sign(dot(k1.dir(),k2.dir()))==1;}
int operator < (line k1,line k2){
    if (sameDir(k1,k2)) return k2.include(k1[0]);
    return compareangle(k1.dir(),k2.dir());
}
int checkpos(line k1,line k2,line k3){return k3.include(getLL(k1,k2));}
vector<line> getHL(vector<line> &L){ // 求半平面交 , 半平面是逆时针方向 , 输出按照逆时针
    sort(L.begin(),L.end()); deque<line> q;
    for (int i=0;i<(int)L.size();i++){
        if (i&&sameDir(L[i],L[i-1])) continue;
        while (q.size()>1&&!checkpos(q[q.size()-2],q[q.size()-1],L[i])) q.pop_back();
        while (q.size()>1&&!checkpos(q[1],q[0],L[i])) q.pop_front();
        q.push_back(L[i]);
    }
    while (q.size()>2&&!checkpos(q[q.size()-2],q[q.size()-1],q[0])) q.pop_back();
    while (q.size()>2&&!checkpos(q[1],q[0],q[q.size()-1])) q.pop_front();
    vector<line>ans; for (int i=0;i<q.size();i++) ans.push_back(q[i]);
    return ans;
}
db closepoint(vector<point>&A,int l,int r){ // 最近点对 , 先要按照 x 坐标排序
    if (r-l<=5){
        db ans=1e20;
        for (int i=l;i<=r;i++) for (int j=i+1;j<=r;j++) ans=min(ans,A[i].dis(A[j]));
        return ans;
    }
    int mid=l+r>>1; db ans=min(closepoint(A,l,mid),closepoint(A,mid+1,r));
    vector<point>B; for (int i=l;i<=r;i++) if (abs(A[i].x-A[mid].x)<=ans) B.push_back(A[i]);
    sort(B.begin(),B.end(),[](point k1,point k2){return k1.y<k2.y;});
    for (int i=0;i<B.size();i++) for (int j=i+1;j<B.size()&&B[j].y-B[i].y<ans;j++) ans=min(ans,B[i].dis(B[j]));
    return ans;
}
int checkposCC(circle k1,circle k2){// 返回两个圆的公切线数量
    if (cmp(k1.r,k2.r)==-1) swap(k1,k2);
    db dis=k1.o.dis(k2.o);  int w1=cmp(dis,k1.r+k2.r),w2=cmp(dis,k1.r-k2.r);
    if (w1>0) return 4; else if (w1==0) return 3; else if (w2>0) return 2;
    else if (w2==0) return 1; else return 0;
}
vector<point> getCL(circle k1,point k2,point k3){ // 沿着 k2->k3 方向给出 , 相切给出两个
    point k=proj(k2,k3,k1.o); db d=k1.r*k1.r-(k-k1.o).abs2();
    if (sign(d)==-1) return {};
    point del=(k3-k2).unit()*sqrt(max((db)0.0,d)); return {k-del,k+del};
}
vector<point> getCC(circle k1,circle k2){// 沿圆 k1 逆时针给出 , 相切给出两个
    int pd=checkposCC(k1,k2); if (pd==0||pd==4) return {};
    db a=(k2.o-k1.o).abs2(),cosA=(k1.r*k1.r+a-k2.r*k2.r)/(2*k1.r*sqrt(max(a,(db)0.0)));
    db b=k1.r*cosA,c=sqrt(max((db)0.0,k1.r*k1.r-b*b));
    point k=(k2.o-k1.o).unit(),m=k1.o+k*b,del=k.turn90()*c;
    return {m-del,m+del};
}
vector<point> TangentCP(circle k1,point k2){// 沿圆 k1 逆时针给出
```

```cpp
    db a=(k2-k1.o).abs(),b=k1.r*k1.r/a,c=sqrt(max((db)0.0,k1.r*k1.r-b*b));
    point k=(k2-k1.o).unit(),m=k1.o+k*b,del=k.turn90()*c;
    return {m-del,m+del};
}
vector<line> TangentoutCC(circle k1,circle k2){
    int pd=checkposCC(k1,k2); if (pd==0) return {};
    if (pd==1){point k=getCC(k1,k2)[0]; return {(line){k,k}};}
    if (cmp(k1.r,k2.r)==0){
        point del=(k2.o-k1.o).unit().turn90().getdel();
        return {(line){k1.o-del*k1.r,k2.o-del*k2.r},(line){k1.o+del*k1.r,k2.o+del*k2.r}};
    } else {
        point p=(k2.o*k1.r-k1.o*k2.r)/(k1.r-k2.r);
        vector<point>A=TangentCP(k1,p),B=TangentCP(k2,p);
        vector<line>ans; for (int i=0;i<A.size();i++) ans.push_back((line){A[i],B[i]});
        return ans;
    }
}
vector<line> TangentinCC(circle k1,circle k2){
    int pd=checkposCC(k1,k2); if (pd<=2) return {};
    if (pd==3){point k=getCC(k1,k2)[0]; return {(line){k,k}};}
    point p=(k2.o*k1.r+k1.o*k2.r)/(k1.r+k2.r);
    vector<point>A=TangentCP(k1,p),B=TangentCP(k2,p);
    vector<line>ans; for (int i=0;i<A.size();i++) ans.push_back((line){A[i],B[i]});
    return ans;
}
vector<line> TangentCC(circle k1,circle k2){
    int flag=0; if (k1.r<k2.r) swap(k1,k2),flag=1;
    vector<line>A=TangentoutCC(k1,k2),B=TangentinCC(k1,k2);
    for (line k:B) A.push_back(k);
    if (flag) for (line &k:A) swap(k[0],k[1]);
    return A;
}
db getarea(circle k1,point k2,point k3){
    // 圆 k1 与三角形 k2 k3 k1.o 的有向面积交
    point k=k1.o; k1.o=k1.o-k; k2=k2-k; k3=k3-k;
    int pd1=k1.inside(k2),pd2=k1.inside(k3);
    vector<point>A=getCL(k1,k2,k3);
    if (pd1>=0){
        if (pd2>=0) return cross(k2,k3)/2;
        return k1.r*k1.r*rad(A[1],k3)/2+cross(k2,A[1])/2;
    } else if (pd2>=0){
        return k1.r*k1.r*rad(k2,A[0])/2+cross(A[0],k3)/2;
    }else {
        int pd=cmp(k1.r,disSP(k2,k3,k1.o));
        if (pd<=0) return k1.r*k1.r*rad(k2,k3)/2;
        return cross(A[0],A[1])/2+k1.r*k1.r*(rad(k2,A[0])+rad(A[1],k3))/2;
    }
}
circle getcircle(point k1,point k2,point k3){
    db a1=k2.x-k1.x,b1=k2.y-k1.y,c1=(a1*a1+b1*b1)/2;
    db a2=k3.x-k1.x,b2=k3.y-k1.y,c2=(a2*a2+b2*b2)/2;
    db d=a1*b2-a2*b1;
    point o=(point){k1.x+(c1*b2-c2*b1)/d,k1.y+(a1*c2-a2*c1)/d};
    return (circle){o,k1.dis(o)};
}
circle getScircle(vector<point> A){
    random_shuffle(A.begin(),A.end());
    circle ans=(circle){A[0],0};
    for (int i=1;i<A.size();i++)
        if (ans.inside(A[i])==-1){
            ans=(circle){A[i],0};
            for (int j=0;j<i;j++)
                if (ans.inside(A[j])==-1){
                    ans.o=(A[i]+A[j])/2; ans.r=ans.o.dis(A[i]);
                    for (int k=0;k<j;k++)
                        if (ans.inside(A[k])==-1)
                            ans=getcircle(A[i],A[j],A[k]);
                }
        }
    return ans;
}
db area(vector<point> A){ // 多边形用 vector<point> 表示 , 逆时针
    db ans=0;
    for (int i=0;i<A.size();i++) ans+=cross(A[i],A[(i+1)%A.size()]);
    return ans/2;
}
int checkconvex(vector<point>A){
    int n=A.size(); A.push_back(A[0]); A.push_back(A[1]);
    for (int i=0;i<n;i++) if (sign(cross(A[i+1]-A[i],A[i+2]-A[i]))==-1) return 0;
    return 1;
}
int contain(vector<point>A,point q){ // 2 内部 1 边界 0 外部
    int pd=0; A.push_back(A[0]);
    for (int i=1;i<A.size();i++){
        point u=A[i-1],v=A[i];
        if (onS(u,v,q)) return 1; if (cmp(u.y,v.y)>0) swap(u,v);
        if (cmp(u.y,q.y)>=0||cmp(v.y,q.y)<0) continue;
        if (sign(cross(u-v,q-v))<0) pd^=1;
```

```
    }
    return pd<<1;
}
vector<point> ConvexHull(vector<point>A,int flag=1){ // flag=0 不严格 flag=1 严格
    int n=A.size(); vector<point>ans(n*2);
    sort(A.begin(),A.end()); int now=-1;
    for (int i=0;i<A.size();i++){
        while (now>0&&sign(cross(ans[now]-ans[now-1],A[i]-ans[now-1]))<flag) now--;
        ans[++now]=A[i];
    } int pre=now;
    for (int i=n-2;i>=0;i--){
        while (now>pre&&sign(cross(ans[now]-ans[now-1],A[i]-ans[now-1]))<flag) now--;
        ans[++now]=A[i];
    } ans.resize(now); return ans;
}
db convexDiameter(vector<point>A){
    int now=0,n=A.size(); db ans=0;
    for (int i=0;i<A.size();i++){
        now=max(now,i);
        while (1){
            db k1=A[i].dis(A[now%n]),k2=A[i].dis(A[(now+1)%n]);
            ans=max(ans,max(k1,k2)); if (k2>k1) now++; else break;
        }
    }
    return ans;
}
vector<point> convexcut(vector<point>A,point k1,point k2){
    // 保留 k1,k2,p 逆时针的所有点
    int n=A.size(); A.push_back(A[0]); vector<point>ans;
    for (int i=0;i<n;i++){
        int w1=clockwise(k1,k2,A[i]),w2=clockwise(k1,k2,A[i+1]);
        if (w1>=0) ans.push_back(A[i]);
        if (w1*w2<0) ans.push_back(getLL(k1,k2,A[i],A[i+1]));
    }
    return ans;
}
int checkPoS(vector<point>A,point k1,point k2){
    // 多边形 A 和直线 ( 线段 )k1->k2 严格相交 , 注释部分为线段
    struct ins{
        point m,u,v;
        int operator < (const ins& k) const {return m<k.m;}
    }; vector<ins>B;
    //if (contain(A,k1)==2||contain(A,k2)==2) return 1;
    vector<point>poly=A; A.push_back(A[0]);
    for (int i=1;i<A.size();i++) if (checkLL(A[i-1],A[i],k1,k2)){
        point m=getLL(A[i-1],A[i],k1,k2);
        if (inmid(A[i-1],A[i],m)/*&&inmid(k1,k2,m)*/) B.push_back((ins){m,A[i-1],A[i]});
    }
    if (B.size()==0) return 0; sort(B.begin(),B.end());
    int now=1; while (now<B.size()&&B[now].m==B[0].m) now++;
    if (now==B.size()) return 0;
    int flag=contain(poly,(B[0].m+B[now].m)/2);
    if (flag==2) return 1;
    point d=B[now].m-B[0].m;
    for (int i=now;i<B.size();i++){
        if (!(B[i].m==B[i-1].m)&&flag==2) return 1;
        int tag=sign(cross(B[i].v-B[i].u,B[i].m+d-B[i].u));
        if (B[i].m==B[i].u||B[i].m==B[i].v) flag+=tag; else flag+=tag*2;
    }
    //return 0;
    return flag==2;
}
int checkinp(point r,point l,point m){
  if (compareangle(l,r)){return compareangle(l,m)&&compareangle(m,r);}
  return compareangle(l,m)||compareangle(m,r);
}
int checkPosFast(vector<point>A,point k1,point k2){ // 快速检查线段是否和多边形严格相交
  if (contain(A,k1)==2||contain(A,k2)==2) return 1; if (k1==k2) return 0;
  A.push_back(A[0]); A.push_back(A[1]);
  for (int i=1;i+1<A.size();i++)
    if (checkLL(A[i-1],A[i],k1,k2)){
      point now=getLL(A[i-1],A[i],k1,k2);
      if (inmid(A[i-1],A[i],now)==0||inmid(k1,k2,now)==0) continue;
      if (now==A[i]){
        if (A[i]==k2) continue;
        point pre=A[i-1],ne=A[i+1];
        if (checkinp(pre-now,ne-now,k2-now)) return 1;
      } else if (now==k1){
        if (k1==A[i-1]||k1==A[i]) continue;
        if (checkinp(A[i-1]-k1,A[i]-k1,k2-k1)) return 1;
      } else if (now==k2||now==A[i-1]) continue;
      else return 1;
    }
  return 0;
}
// 拆分凸包成上下凸壳 凸包尽量都随机旋转一个角度来避免出现相同横坐标
// 尽量特判只有一个点的情况 凸包逆时针
void getUDP(vector<point>A,vector<point>&U,vector<point>&D){
```

```cpp
    db l=1e100,r=-1e100;
    for (int i=0;i<A.size();i++) l=min(l,A[i].x),r=max(r,A[i].x);
    int wherel,wherer;
    for (int i=0;i<A.size();i++) if (cmp(A[i].x,l)==0) wherel=i;
    for (int i=A.size();i;i--) if (cmp(A[i-1].x,r)==0) wherer=i-1;
    U.clear(); D.clear(); int now=wherel;
    while (1){D.push_back(A[now]); if (now==wherer) break; now++; if (now>=A.size()) now=0;}
    now=wherel;
    while (1){U.push_back(A[now]); if (now==wherer) break; now--; if (now<0) now=A.size()-1;}
}
// 需要保证凸包点数大于等于 3,2 内部 ,1 边界 ,0 外部
int containCoP(const vector<point>&U,const vector<point>&D,point k){
    db lx=U[0].x,rx=U[U.size()-1].x;
    if (k==U[0]||k==U[U.size()-1]) return 1;
    if (cmp(k.x,lx)==-1||cmp(k.x,rx)==1) return 0;
    int where1=lower_bound(U.begin(),U.end(),(point){k.x,-1e100})-U.begin();
    int where2=lower_bound(D.begin(),D.end(),(point){k.x,-1e100})-D.begin();
    int w1=clockwise(U[where1-1],U[where1],k),w2=clockwise(D[where2-1],D[where2],k);
    if (w1==1||w2==-1) return 0; else if (w1==0||w2==0) return 1; return 2;
}
// d 是方向 , 输出上方切点和下方切点
pair<point,point> getTangentCow(const vector<point> &U,const vector<point> &D,point d){
    if (sign(d.x)<0||(sign(d.x)==0&&sign(d.y)<0)) d=d*(-1);
    point whereU,whereD;
    if (sign(d.x)==0) return mp(U[0],U[U.size()-1]);
    int l=0,r=U.size()-1,ans=0;
    while (l<r){int mid=l+r>>1; if (sign(cross(U[mid+1]-U[mid],d))<=0) l=mid+1,ans=mid+1; else r=mid;}
    whereU=U[ans]; l=0,r=D.size()-1,ans=0;
    while (l<r){int mid=l+r>>1; if (sign(cross(D[mid+1]-D[mid],d))>=0) l=mid+1,ans=mid+1; else r=mid;}
    whereD=D[ans]; return mp(whereU,whereD);
}
// 先检查 contain, 逆时针给出
pair<point,point> getTangentCoP(const vector<point>&U,const vector<point>&D,point k){
    db lx=U[0].x,rx=U[U.size()-1].x;
    if (k.x<lx){
        int l=0,r=U.size()-1,ans=U.size()-1;
        while (l<r){int mid=l+r>>1; if (clockwise(k,U[mid],U[mid+1])==1) l=mid+1; else ans=mid,r=mid;}
        point w1=U[ans]; l=0,r=D.size()-1,ans=D.size()-1;
        while (l<r){int mid=l+r>>1; if (clockwise(k,D[mid],D[mid+1])==-1) l=mid+1; else ans=mid,r=mid;}
        point w2=D[ans]; return mp(w1,w2);
    } else if (k.x>rx){
        int l=1,r=U.size(),ans=0;
        while (l<r){int mid=l+r>>1; if (clockwise(k,U[mid],U[mid-1])==-1) r=mid; else ans=mid,l=mid+1;}
        point w1=U[ans]; l=1,r=D.size(),ans=0;
        while (l<r){int mid=l+r>>1; if (clockwise(k,D[mid],D[mid-1])==1) r=mid; else ans=mid,l=mid+1;}
        point w2=D[ans]; return mp(w2,w1);
    } else {
        int where1=lower_bound(U.begin(),U.end(),(point){k.x,-1e100})-U.begin();
        int where2=lower_bound(D.begin(),D.end(),(point){k.x,-1e100})-D.begin();
        if ((k.x==lx&&k.y>U[0].y)||(where1&&clockwise(U[where1-1],U[where1],k)==1)){
            int l=1,r=where1+1,ans=0;
            while (l<r){int mid=l+r>>1; if (clockwise(k,U[mid],U[mid-1])==1) ans=mid,l=mid+1; else r=mid;}
            point w1=U[ans]; l=where1,r=U.size()-1,ans=U.size()-1;
            while (l<r){int mid=l+r>>1; if (clockwise(k,U[mid],U[mid+1])==1) l=mid+1; else ans=mid,r=mid;}
            point w2=U[ans]; return mp(w2,w1);
        } else {
            int l=1,r=where2+1,ans=0;
            while (l<r){int mid=l+r>>1; if (clockwise(k,D[mid],D[mid-1])==-1) ans=mid,l=mid+1; else r=mid;}
            point w1=D[ans]; l=where2,r=D.size()-1,ans=D.size()-1;
            while (l<r){int mid=l+r>>1; if (clockwise(k,D[mid],D[mid+1])==-1) l=mid+1; else ans=mid,r=mid;}
            point w2=D[ans]; return mp(w1,w2);
        }
    }
}
struct P3{
    db x,y,z;
    P3 operator + (P3 k1){return (P3){x+k1.x,y+k1.y,z+k1.z};}
    P3 operator - (P3 k1){return (P3){x-k1.x,y-k1.y,z-k1.z};}
    P3 operator * (db k1){return (P3){x*k1,y*k1,z*k1};}
    P3 operator / (db k1){return (P3){x/k1,y/k1,z/k1};}
    db abs2(){return x*x+y*y+z*z;}
    db abs(){return sqrt(x*x+y*y+z*z);}
    P3 unit(){return (*this)/abs();}
    int operator < (const P3 k1) const{
        if (cmp(x,k1.x)!=0) return x<k1.x;
        if (cmp(y,k1.y)!=0) return y<k1.y;
```

```
        return cmp(z,k1.z)==-1;
    }
    int operator == (const P3 k1){
        return cmp(x,k1.x)==0&&cmp(y,k1.y)==0&&cmp(z,k1.z)==0;
    }
    void scan(){
        double k1,k2,k3; scanf("%lf%lf%lf",&k1,&k2,&k3);
        x=k1; y=k2; z=k3;
    }
};
P3 cross(P3 k1,P3 k2){return
  ↪ (P3){k1.y*k2.z-k1.z*k2.y,k1.z*k2.x-k1.x*k2.z,k1.x*k2.y-k1.y*k2.x};}
db dot(P3 k1,P3 k2){return k1.x*k2.x+k1.y*k2.y+k1.z*k2.z;}
//p=(3,4,5),l=(13,19,21),theta=85 ans=(2.83,4.62,1.77)
P3 turn3D(db k1,P3 l,P3 p){
    l=l.unit(); P3 ans; db c=cos(k1),s=sin(k1);
    ans.x=p.x*(l.x*l.x*(1-c)+c)+p.y*(l.x*l.y*(1-c)-l.z*s)+p.z*(l.x*l.z*(1-c)+l.y*s);
    ans.y=p.x*(l.x*l.y*(1-c)+l.z*s)+p.y*(l.y*l.y*(1-c)+c)+p.z*(l.y*l.z*(1-c)-l.x*s);
    ans.z=p.x*(l.x*l.z*(1-c)-l.y*s)+p.y*(l.y*l.z*(1-c)+l.x*s)+p.z*(l.x*l.x*(1-c)+c);
    return ans;
}
typedef vector<P3> VP;
typedef vector<VP> VVP;
db Acos(db x){return acos(max(-(db)1,min(x,(db)1)));}
// 球面距离 , 圆心原点 , 半径 1
db Odist(P3 a,P3 b){db r=Acos(dot(a,b)); return r;}
db r; P3 rnd;
vector<db> solve(db a,db b,db c){
    db r=sqrt(a*a+b*b),th=atan2(b,a);
    if (cmp(c,-r)==-1) return {0};
    else if (cmp(r,c)<=0) return {1};
    else {
        db tr=pi-Acos(c/r); return {th+pi-tr,th+pi+tr};
    }
}
vector<db> jiao(P3 a,P3 b){
    // dot(rd+x*cos(t)+y*sin(t),b) >= cos(r)
    if (cmp(Odist(a,b),2*r)>0) return {0};
    P3 rd=a*cos(r),z=a.unit(),y=cross(z,rnd).unit(),x=cross(y,z).unit();
    vector<db> ret =
  ↪ solve(-(dot(x,b)*sin(r)),-(dot(y,b)*sin(r)),-(cos(r)-dot(rd,b)));
    return ret;
}
db norm(db x,db l=0,db r=2*pi){ // change x into [l,r)
    while (cmp(x,l)==-1) x+=(r-l); while (cmp(x,r)>=0) x-=(r-l);
    return x;
}
db disLP(P3 k1,P3 k2,P3 q){
    return (cross(k2-k1,q-k1)).abs()/(k2-k1).abs();
}
db disLL(P3 k1,P3 k2,P3 k3,P3 k4){
    P3 dir=cross(k2-k1,k4-k3); if (sign(dir.abs())==0) return disLP(k1,k2,k3);
    return fabs(dot(dir.unit(),k1-k2));
}
VP getFL(P3 p,P3 dir,P3 k1,P3 k2){
    db a=dot(k2-p,dir),b=dot(k1-p,dir),d=a-b;
    if (sign(fabs(d))==0) return {};
    return {(k1*a-k2*b)/d};
}
VP getFF(P3 p1,P3 dir1,P3 p2,P3 dir2){// 返回一条线
    P3 e=cross(dir1,dir2),v=cross(dir1,e);
    db d=dot(dir2,v); if (sign(abs(d))==0) return {};
    P3 q=p1+v*dot(dir2,p2-p1)/d; return {q,q+e};
}
// 3D Covex Hull Template
db getV(P3 k1,P3 k2,P3 k3,P3 k4){ // get the Volume
    return dot(cross(k2-k1,k3-k1),k4-k1);
}
db rand_db(){return 1.0*rand()/RAND_MAX;}
VP convexHull2D(VP A,P3 dir){
    P3 x={(db)rand(),(db)rand(),(db)rand()}; x=x.unit();
    x=cross(x,dir).unit(); P3 y=cross(x,dir).unit();
    P3 vec=dir.unit()*dot(A[0],dir);
    vector<point>B;
    for (int i=0;i<A.size();i++) B.push_back((point){dot(A[i],x),dot(A[i],y)});
    B=ConvexHull(B); A.clear();
    for (int i=0;i<B.size();i++) A.push_back(x*B[i].x+y*B[i].y+vec);
    return A;
}
namespace CH3{
    VVP ret; set<pair<int,int> >e;
    int n; VP p,q;
    void wrap(int a,int b){
        if (e.find({a,b})==e.end()){
            int c=-1;
            for (int i=0;i<n;i++) if (i!=a&&i!=b){
                if (c==-1||sign(getV(q[c],q[a],q[b],q[i]))>0) c=i;
            }
            if (c!=-1){
                ret.push_back({p[a],p[b],p[c]});
```

```cpp
                e.insert({a,b}); e.insert({b,c}); e.insert({c,a});
                wrap(c,b); wrap(a,c);
            }
        }
    }
    VVP ConvexHull3D(VP _p){
        p=q=_p; n=p.size();
        ret.clear(); e.clear();
        for (auto &i:q) i=i+(P3){rand_db()*1e-4,rand_db()*1e-4,rand_db()*1e-4};
        for (int i=1;i<n;i++) if (q[i].x<q[0].x) swap(p[0],p[i]),swap(q[0],q[i]);
        for (int i=2;i<n;i++) if
        ↪ ((q[i].x-q[0].x)*(q[1].y-q[0].y)>(q[i].y-q[0].y)*(q[1].x-q[0].x))
        ↪ swap(q[1],q[i]),swap(p[1],p[i]);
        wrap(0,1);
        return ret;
    }
}
VVP reduceCH(VVP A){
    VVP ret; map<P3,VP> M;
    for (VP nowF:A){
        P3 dir=cross(nowF[1]-nowF[0],nowF[2]-nowF[0]).unit();
        for (P3 k1:nowF) M[dir].pb(k1);
    }
    for (pair<P3,VP> nowF:M) ret.pb(convexHull2D(nowF.se,nowF.fi));
    return ret;
}
//  把一个面变成 ( 点 , 法向量 ) 的形式
pair<P3,P3> getF(VP F){
    return mp(F[0],cross(F[1]-F[0],F[2]-F[0]).unit());
}
// 3D Cut 保留 dot(dir,x-p)>=0 的部分
VVP ConvexCut3D(VVP A,P3 p,P3 dir){
    VVP ret; VP sec;
    for (VP nowF: A){
        int n=nowF.size(); VP ans; int dif=0;
        for (int i=0;i<n;i++){
            int d1=sign(dot(dir,nowF[i]-p));
            int d2=sign(dot(dir,nowF[(i+1)%n]-p));
            if (d1>=0) ans.pb(nowF[i]);
            if (d1*d2<0){
                P3 q=getFL(p,dir,nowF[i],nowF[(i+1)%n])[0];
                ans.push_back(q); sec.push_back(q);
            }
            if (d1==0) sec.push_back(nowF[i]); else dif=1;

        ↪ dif|=(sign(dot(dir,cross(nowF[(i+1)%n]-nowF[i],nowF[(i+1)%n]-nowF[i])))==-1);
        }
        if (ans.size()>0&&dif) ret.push_back(ans);
    }
    if (sec.size()>0) ret.push_back(convexHull2D(sec,dir));
    return ret;
}
db vol(VVP A){
    if (A.size()==0) return 0; P3 p=A[0][0]; db ans=0;
    for (VP nowF:A)
        for (int i=2;i<nowF.size();i++)
            ans+=abs(getV(p,nowF[0],nowF[i-1],nowF[i]));
    return ans/6;
}
VVP init(db INF) {
    VVP pss(6,VP(4));
    pss[0][0] = pss[1][0] = pss[2][0] = {-INF, -INF, -INF};
    pss[0][3] = pss[1][1] = pss[5][2] = {-INF, -INF, INF};
    pss[0][1] = pss[2][3] = pss[4][2] = {-INF, INF, -INF};
    pss[0][2] = pss[5][3] = pss[4][1] = {-INF, INF, INF};
    pss[1][3] = pss[2][1] = pss[3][2] = {INF, -INF, -INF};
    pss[1][2] = pss[5][1] = pss[3][3] = {INF, -INF, INF};
    pss[2][2] = pss[4][3] = pss[3][1] = {INF, INF, -INF};
    pss[5][0] = pss[4][0] = pss[3][0] = {INF, INF, INF};
    return pss;
}
```

弦图相关

1. 团数 $\leq$ 色数，弦图团数 $=$ 色数

2. 设 $next(v)$ 表示 $N(v)$ 中最前的点．令 w* 表示所有满足 $A \in B$ 的 w 中最后的一个点，判断 $v \cup N(v)$ 是否为极大团，只需判断是否存在一个 w, 满足 $Next(w) = v$ 且 $|N(v)| + 1 \leq |N(w)|$ 即可．

3. 最小染色：完美消除序列从后往前依次给每个点染色，给每个点染上可以染的最小的颜色

4. 最大独立集：完美消除序列从前往后能选就选

5. 弦图最大独立集数 $=$ 最小团覆盖数，最小团覆盖：设最大独立集为 $\{p_1, p_2, \ldots, p_t\}$, 则 $\{p_1 \cup N(p_1), \ldots, p_t \cup N(p_t)\}$ 为最小团覆盖

综合

二分图　定理 1: 最小覆盖数 $=$ 最大匹配数

定理 2: 最大独立集 S 与 最小覆盖集 T 互补

算法:

1. 做最大匹配， 没有匹配的空闲点 $\in S$
2. 如果 $u \in S$ 那么 u 的临点必然属于 T
3. 如果一对匹配的点中有一个属于 T 那么另外一个属于 S
4. 还不能确定的， 把左子图的放入 S，右子图放入 T

算法结束

上下界流　上下界无源汇可行流 ： 不用添 $T->S$， 判断是否流量平衡
上下界有源汇可行流 ： 添 $T\to S$（ 下界 0，上界 ∞）， 判断是否流量平衡
上下界最小流 ： 不添 $T\to S$ 先流一遍 ，再添 $T\to S$（ 下界 0，上界 ∞）在残图上流一遍 ，答案为 $S\to T$ 的流量值
上下界最大流 ： 添 $T\to S$（ 下界 0，上界 ∞）流一遍，再在残图上流一遍S到T的最大流，答案为前者的 $S\to T$ 的值 ＋ 残图中 $S\to T$ 的最大流 （ 不删那条边的话，最后的最大流就是答案 ）

最大流对偶　考虑最大费用循环流的标准线性规划建模：

Maximize: $\sum_{i\in E} cost_i \cdot f_i$

- □ 对每条弧i有 $0\le f_i\le cap_i$ ，cap_i 表示这条弧的容量，$f_i\ge 0$。
- □ 对于每个点x有流量平衡： $\sum_{u_i=x} f_i - \sum_{v_i=x} f_i = 0$

共有$|V|+|E|$个限制，对偶后，设前$|V|$个限制对应的变量为a_i，后$|E|$个限制对应的变量为d_i：

Minimize: $\sum_{i\in E} cap_i \cdot d_i$

- 对每条弧i有 $a_{v_i} - a_{u_i} + d_i \ge cost_i$。
- a_x无限制，$d_i\ge 0$。
  - $min \ge --> max \le$

所以，比如有很多变量然后给定一些差分后的不等式然后可以花费代价让一个不等式“放宽”，目标总代价最小的模型，都是最大费用流的对偶。

类欧几里得

- $f(a,b,c,n)=\sum_{i=0}^{n}\lfloor\frac{ai+b}{c}\rfloor$
- $m=\lfloor\frac{an+b}{c}\rfloor, f(a,b,c,n)=nm-f(c,c-b-1,a,m-1)$

拟阵　1、求最小权基，贪心；
2、求两个拟阵(M_1,I_1)和(M_2,I_2)的最小权拟阵交，从空集开始每次增加一个元素，
假设当前集合为A，建图：
如果x不属于A，$A+\{x\}\in I_1$，连边S->x，边权为x的权值；
如果x不属于A，$A+\{x\}\in I_2$，连边x->T，边权为0；
如果x不属于A， y属于A，$A-\{y\}+\{x\}\in I_2$，连边x->y,边权为y的权值的相反数；
如果x不属于A， y属于A，$A-\{y\}+\{x\}\in I_1$，连边y->x，边权为x的权值；
找出S->T的最短路，把路径上每个点的是否在集合里取反。
3、把S分解为最少的拟阵的并：
最小值为$\max\lceil\frac{|S|}{r(|S|)}\rceil$
每次增加一个元素x，每个当前的等价类A_i连边$S->A_i$。
如果y不属于A_i，$A_i+\{y\}\in I$，连边$A_i->y$。
如果y不属于A_i，z属于A_i，$A_i-\{z\}+\{y\}\in I$，连边$y->z$。

染色多项式

$$\text{number of acyclic orientations of G is } (-1)^{|V(G)|}P(G,-1)$$

$$\text{Cycle } P(C_n,t)=(t-1)^n+(-1)^n(t-1)$$

Petersen graph $P(P_5,t)=t(t-1)(t-2)(t^7-12t^6+67t^5-230t^4+529t^3-814t^2+775t-$

伯努利数

$$\sum_{k=0}^{m-1}k^n=\frac{1}{n+1}\sum_{k=0}^{n}\binom{n+1}{k}B_k m^{n+1-k}$$

$$\sum_{j=0}^{m}\binom{m+1}{j}B_j=0 \qquad \frac{B_{m+p-1}}{m+p-1}\equiv\frac{B_m}{m}(\mod p)$$

高维单位球

$$A(d)=\frac{2\pi^{\frac{d}{2}}}{\Gamma(\frac{d}{2})}, V(d)=\frac{1}{d}A(d)$$

基本形

椭圆　标准形 $\frac{x^2}{a^2}+\frac{y^2}{b^2}=1$，离心率 $e=\frac{c}{a}, c=\sqrt{a^2-b^2}$，焦点参数 $p=\frac{b^2}{a}$

椭圆上(x,y)处曲率半径 $R=a^2b^2(\frac{x^2}{a^4}+\frac{y^2}{b^4})^{\frac{3}{2}}=\frac{(r_1r_2)^{\frac{3}{2}}}{ab}$，其中$r_i$为到焦点$F_i$距离

点$A(a,0)$，$M(x,y)$ 则扇形面积 $S_{OAM}=\frac{1}{2}ab\arccos\frac{x}{a}$ 弧长

$$L_{AM}=a\int_0^{\arccos\frac{x}{a}}\sqrt{1-e^2cos^2t}dt=a\int_{\arccos\frac{x}{a}}^{\frac{\pi}{2}}\sqrt{1-e^2\sin^2t}dt$$

周长 $L=2a\pi[1-(\frac{1}{2})^2e^2-(\frac{1\times3}{2\times4})^2\frac{e^4}{3}-\dots]$ 极坐标方程 $r^2=\frac{b^2a^2}{b^2\cos^2\theta+a^2\sin^2\theta}$

抛物线　标准形 $y^2=2px$，曲率半径 $R=((p+2x)^{3/2})/\sqrt{p}$，其中$r_i$为到焦点$F_i$距离

点$A(a,0)$，$M(x,y)$ 则扇形面积 $S_{OAM}=\frac{1}{2}ab\arccos\frac{x}{a}$ 弧长

$$L_{OM}=\frac{p}{2}[\sqrt{\frac{2x}{p}(1+\frac{2x}{p})}+\ln(\frac{2x}{p}+\sqrt{1+\frac{2x}{p}})]$$

重心 半径r圆心角θ的扇形重心与圆心距离 $\frac{4r}{3\theta}\sin\frac{\theta}{2}$

半径r圆心角θ的圆弧重心与圆心距离 $\frac{4r}{3\theta-3\sin\theta}\sin^3\frac{\theta}{2}$

椭圆上半部分重心与圆心距离 $\frac{4}{3\pi}b$

树的计数 若$n+1$个点的有根树总数为a_{n+1}，无根树总数为b_{n+1}，$a_i=\{1,1,2,4,9,20,286,1842\ldots\}$

$$S_{n,j}=\sum_{i=1}^{n/j}a_{n+1-ij}=S_{n-j,j}+a_{n+1-j}\qquad a_{n+1}=\frac{1}{n}\sum_{j=1}^{n}ja_jS_{n,j}$$

$$b_{2k+1}=a_n-\sum_{i=1}^{n/2}a_ia_{n-i}\qquad b_{2k}=a_n-\sum_{i=1}^{n/2}a_ia_{n-i}+\frac{1}{2}a_{n/2}(a_{n/2}+1)$$

组合公式

$$\sum_{k=1}^{n}k^5=\frac{1}{12}n^2(n+1)^2(2N^2+2n-1)\qquad \sum_{k=1}^{n}k^4=\frac{1}{30}n(n+1)(2n+1)(3n^2+3n-1)$$

$$\text{限位排列}Ans=\sum_{i=0}^{n}(-1)^k*r_k*(n-i)!$$

其中r_k表示把k个物品放在不能放的位置上使得每行每列至多一个的方案数

三角公式

$$\sin(\alpha\pm\beta)=\sin\alpha\cos\beta\pm\cos\alpha\sin\beta\qquad\cos(\alpha\pm\beta)=\cos\alpha\cos\beta\mp\sin\alpha\sin\beta$$

$$\tan(\alpha\pm\beta)=\frac{\tan\alpha\pm\tan\beta}{1\mp\tan\alpha\tan\beta}\qquad\tan(\alpha)\pm\tan(\beta)=\frac{\sin(\alpha\pm\beta)}{\cos\alpha\cos\beta}$$

$$\sin(n\alpha)=n\cos^{n-1}\alpha\sin\alpha-\binom{n}{3}\cos^{n-3}\alpha\sin^3\alpha+\binom{n}{5}\cos^{n-5}\alpha\sin^5\alpha-\ldots$$

$$\cos(n\alpha)=\cos^n\alpha\sin\alpha-\binom{n}{2}\cos^{n-2}\alpha\sin^2\alpha+\binom{n}{4}\cos^{n-4}\alpha\sin^4\alpha-\ldots$$

反演

$$a_n=\sum_{k=0}^{n}C_n^kb_k,\qquad b_n=\sum_{k=0}^{n}(-1)^{k+n}C_n^ka_k$$

$$a_n=\sum_{k=n}^{\inf}C_k^nb_k,\qquad b_n=\sum_{k=n}^{\inf}(-1)^{k+n}C_k^na_k$$

$$a_n=\sum_{k=0}^{n}C_{n+p}^{k+p}b_k,\qquad b_n=\sum_{k=n}^{\inf}(-1)^{k+n}C_{n+p}^{k+p}a_k$$

$$a_n=\sum_{k=n}^{\inf}C_{k+p}^{n+p}b_k,\qquad b_n=\sum_{k=n}^{\inf}(-1)^{k+n}C_{k+p}^{n+p}a_k$$

$$f(n)=\sum_{d|n}g(d),\qquad g(n)=\sum_{d|n}\mu(d)f(\frac{n}{d})$$

杜教筛 $S(n)=\sum_{i=1}^{n}f(i)$

$$g(1)S(n)=\sum_{i=1}^{n}(f*g)(i)-\sum_{i=2}^{n}g(i)S(\lfloor\frac{n}{i}\rfloor)$$

$S(n)=\sum_{i=1}^{n}(f\cdot g)(i)$，$g(x)$ 为完全积性函数。有：

$$S(n)=\sum_{i=1}^{n}[(f*1)\cdot g](i)-\sum_{i=2}^{n}S(\lfloor\frac{n}{i}\rfloor)g(i)$$

$S(n)=\sum_{i=1}^{n}(f*g)(i)$。有：

$$S(n)=\sum_{i=1}^{n}g(i)\sum_{ij\le n}(f*1)(j)-\sum_{i=2}^{n}S(\lfloor\frac{n}{i}\rfloor)$$