Standard Code Library

huicpc0860

2011 年 9 月 19 日

Contents

Chapter 1

Data Structures

1.1 回文子串 Manacher(O(n)) 算法

```
#define N 400010
char s[N];
int p[2*N];

bool cmp(int a,int b){
    if(a<0)return 0;
    if(((a|b)&1)==0)return 1;
    return s[a>>1]==s[b>>1];
}

void mana(int n) {//#a#
    int mx = 0, id;
    for (int i = 0; i <= n; i++) {
        p[i] = i < mx ? min(p[(id << 1) - i], mx - i) : 1;
        while (cmp(i - p[i],i + p[i]))p[i]++;
        if (i + p[i] > mx) {
            mx = i + p[i];
            id = i;
        }
    }
}
```

1.1.1 查询区间 [l,r] 的最长回文子串

二分 +RMQ

```
struct RMQ {//下标由开始0
    int R[32], dp[20][N], n;

    RMQ() {
        for (int i = 0; R[i] = 1 << i, R[i] < N; i++);
    }

    void RM(int *a, int n) {
        for (int i = 0; i < n; i++)dp[0][i] = a[i];
        for (int i = 1; R[i] <= n; i++)
            for (int j = 0; j + R[i] <= n; j++)
                dp[i][j] = max(dp[i - 1][j], dp[i - 1][j + R[i - 1]]);
    }

    int Q(int s, int e) {
        int i = 0, h = (e - s + 1) >> 1;
        while (R[i] <= h)i++;
        return max(dp[i][e - R[i] + 1], dp[i][s]);
    }
} rmq;

int main() {
    int T,n;
    scanf("%d", &T);
    while (T--) {
        scanf("%s", s);
                n = strlen(s) << 1;
        mana(n);
        rmq.RM(p, n);
        int q;
        scanf("%d", &q);
        while (q--) {
            int a, b;
            scanf("%d%d", &a, &b);
            a = (a << 1) - 2;
            b = (b << 1);
            int l = 0, r = (b - a) / 2 + 1;
            while (l < r) {
                int m = (l + r) >> 1;
                if (rmq.Q(a + m, b - m) - 1 < m)r = m;
```

```
41                  else l = m + 1;
42              }
43              printf("%d\n", l - 1);
44          }
45      }
46  }
```

1.1.2 求不同回文子串的个数

不同串长的冲突问题，双关键字 hash。

```
1  #include <cstdio>
2  #include<algorithm>
3  #include <set>
4  using namespace std;
5  typedef long long LL;
6  #define clr(a) memset(a,0,sizeof(a))
7  #define N 100100
8  #define mod 100000007
9  char s[N];
10 int n,cnt;
11 LL c[N],z[N];
12 int p[N<<1];
13 set<int>h[N];
14 inline void judge(int a,int b){
15     if(((a|b)&1)==0)return;
16     a>>=1,b>>=1;
17     int hash=(c[b+1]-c[a]+mod)*z[n-a]%mod;
18     if(h[b-a].size()==0||h[b-a].find(hash)==h[b-a].end()){
19         h[b-a].insert(hash);
20         cnt++;
21     }
22 }
23 inline bool cmp(int a,int b){
24     if(a<0)return 0;
25     if(((a|b)&1)==0)return 1;
26     return s[a>>1]==s[b>>1];
27 }
28 void mana(int n){
29     int mx=0,id;
30     for(int i=0;i<n;i++){
31         p[i]=i<mx?min(p[(id<<1)-i],mx-i):1;
32         judge(i-p[i]+1,i+p[i]-1);
33         while(cmp(i-p[i],i+p[i])){
34             judge(i-p[i],i+p[i]);
35             p[i]++;
36         }
37         if(i+p[i]>mx){
38             mx=i+p[i];
39             id=i;
40         }
41     }
42 }
43 int main(){
44     z[0]=1;
45     z[1]=29;
46     for(int i=2;i<N;i++)z[i]=(z[i-1]*z[1])%mod;
47     int T,cas=1;
48     scanf("%d\n",&T);
49     while(T--){
50         for(int i=0;i<n;i++)h[i].clear();
51         n=0;
52         while(s[n]=getchar(),s[n]!='\n')n++;
53         cnt=0;
54         c[0]=0;
55         for(int i=0;i<n;i++){
56             c[i+1]=((s[i]-'a'+1)*z[i]+c[i])%mod;
```

```
57          }
58          mana(n<<1);
59          printf("Case␣#%d:␣%d\n",cas++,cnt);
60      }
61      return 0;
62  }
```

1.2 HASH

hash 值一般可以到 int，保证相乘不超 LL，bool 数组不能表示可以用 set
不同长度的字符串 hash 冲突很大，可以考虑用加入长度用双关键字 hash。
比如 set[len].insert(hash) 或 set.insert(pt(len,hash))

1.2.1 矩阵二维 hash

n*m 的矩阵按 k*k 的小块 hash

```
1  LL d[1100][1100];
2  LL p1[110];
3  LL p2[110];
4  LL mod1=1000007,mod2=1000000009LL;
5  int n,m,k;
6  int init(){
7      p1[0]=p2[0]=1;
8      p1[1]=23;
9      p2[1]=29;
10     for(int i=2;i<110;i++){
11         p1[i]=(p1[i-1]*p1[1])%mod1;
12         p2[i]=(p2[i-1]*p2[1])%mod2;
13     }
14 }
15 void hush(LL a[][1100],LL b[][1100],int n,int m,int k){
16     for(int i=0;i<n;i++)
17         for(int j=0;j<=m-k;j++)
18             if(j){
19                 d[i][j]=(((d[i][j-1]-a[i][j-1]*p1[k-1])*p1[1]+a[i][j+k-1])%
                       mod1+mod1)%mod1;
20             }else{
21                 d[i][0]=0;
22                 for(int r=0;r<k;r++){
23                     d[i][0]+=(a[i][r]*p1[k-r-1])%mod1;
24                     d[i][0]%=mod1;
25                 }
26             }
27     for(int j=0;j<=m-k;j++)
28         for(int i=0;i<=n-k;i++)
29             if(i){
30                 b[i][j]=(((b[i-1][j]-d[i-1][j]*p2[k-1])*p2[1]+d[i+k-1][j])%
                       mod2+mod2)%mod2;
31             }else{
32                 b[0][j]=0;
33                 for(int r=0;r<k;r++){
34                     b[0][j]+=(d[r][j]*p2[k-r-1])%mod2;
35                     b[0][j]%=mod2;
36                 }
37             }
38 }
```

1.3 AC 自动机

有矩阵乘法也不要忘记 DP
在自己构造串的时候可能因为具体的原因不会超内存。
注意初始化 map，ID=0,newnode(),bfs()

```
#define clr(a) memset(a,0,sizeof(a))
#define N 100
#define D 4
int ID,next[N][D],fail[N],type[N],q[N],map[128];

inline int newnode(){
    clr(next[ID]);
    fail[ID]=type[ID]=0;
    return ID++;
}
inline void insert(char *s){
    int p=0;
    for(int c;*s;p=next[p][c],s++){
        c=map[*s];
        if(!next[p][c])next[p][c]=newnode();
    }
    type[p]=1;
}
void bfs(){
    int *s=q,*e=q;
    *e++=0;
    while(s!=e)
        for(int i=0,p=*s++;i<D;i++)
            if(next[p][i]){
                int t=next[p][i];
                *e++=t;
                if(p){
                    fail[t]=next[fail[p]][i];
                    type[t]|=type[fail[t]];
                }
            }else next[p][i]=next[fail[p]][i];
}
```

1.3.1 包含至少两个关键串的个数，矩阵乘法

每到达一次转移到新的字典树上

```
//cnt[t]+=cnt[fail[t]];
int a[N][N],b[N][N],c[N][N];
void make(int a[][N],int b[][N]){
    clr(c);
    for(int i=0;i<n;i++)
        for(int j=0;j<n;j++)
            for(int k=0;k<n;k++){
                c[i][j]+=a[i][k]*b[k][j];
                c[i][j]%=mod;
            }
    for(int i=0;i<n;i++)
        for(int j=0;j<n;j++)
            a[i][j]=c[i][j];
}
int main(){
    map['A']=0;
    map['T']=1;
    map['C']=2;
    map['G']=3;
    int L;
    while(~scanf("%d%d",&n,&L)){
        ID=0;
        newnode();
        for(int i=0;i<n;i++){
            scanf("%s",s);
            insert(s);
        }
        bfs();
        clr(a);
        clr(b);
```

```
        for(int i=0;i<ID;i++){
            for(int j=0;j<D;j++){
                int k=next[i][j];
                a[i+ID+ID][k+ID+ID]++;
                if(cnt[k])a[i+ID][k+ID+ID]++;
                else a[i+ID][k+ID]++;
                if(cnt[i])continue;
                if(cnt[k]){
                    if(cnt[k]>1)a[i][ID+ID+k]++;
                    else a[i][ID+k]++;
                }else  a[i][k]++;
            }
        }
        n=ID+ID+ID;
        for(int i=0;i<n;i++)b[i][i]=1;
        while(L){
            if(L&1)make(b,a);
            make(a,a);
            L>>=1;
        }
        int ans=0;
        for(int i=0;i<ID;i++){
            ans+=b[0][ID+ID+i];
            ans%=mod;
        }
        printf("%d\n",ans);
    }
    return 0;
}
```

1.3.2 最长被包含字符串

```
void bfs(){
    int *s=q,*e=q;
    *e++=0;
    while(s!=e)
        for(int i=0,p=*s++;i<D;i++)
            if(next[p][i]){
                int t=next[p][i];
                *e++=t;
                if(p){
                    fail[t]=next[fail[p]][i];
                    f[t]=max(f[p],f[fail[t]])+type[t];
                    type[t]|=type[fail[t]];
                }else f[t]=type[t];
                ans=max(ans,f[t]);
            }else next[p][i]=next[fail[p]][i];
}
int main(){
    for(int i=0;i<D;i++)map[i+'a']=i;
    int n;
    while(scanf("%d",&n),n){
        getchar();
        memset(f,0,sizeof(f));
        ID=0;
        newnode();
        ans=0;
        while(n--){
            gets(s);
            insert(s);
        }
        bfs();
        printf("%d\n",ans);
    }
    return 0;
}
```

1.3.3 给定字母个数给定包括最多给定串的排列

任意进制的写法

```
char s[100];
int f[N][11*11*11*11],c[4];
void Max(int &a,int b){
    if(b>a)a=b;
}
int main(){
    map['A']=0;
    map['C']=1;
    map['G']=2;
    map['T']=3;
    int n,cas=1;
    while(scanf("%d",&n),n){
        ID=0;
        newnode();
        for(int i=0;i<n;i++){
            scanf("%s",s);
            insert(s);
        }
        bfs();
        scanf("%s",s);
        n=strlen(s);
        int t[4]={};
        c[0]=1;
        for(int i=0;i<n;i++)
            t[map[s[i]]]++;
        for(int i=1;i<5;i++)
            c[i]=c[i-1]*(t[i-1]+1);
        memset(f,-1,sizeof(f));
        f[0][c[4]-1]=0;
        for(int i=c[4]-1;i>0;i--){
            for(int j=0;j<ID;j++){
                if(f[j][i]<0)continue;
                for(int k=0;k<D;k++)
                    if(i%c[k+1]/c[k])Max(f[next[j][k]][i-c[k]],f[j][i]+cnt[
                        next[j][k]]);
            }
        }
        int ans=0;
        for(int i=0;i<ID;i++)Max(ans,f[i][0]);
        printf("Case␣%d:␣%d\n",cas++,ans);
    }
    return 0;
}
```

1.3.4 用到 fail 指针转移的 dp

有用到 fail 指针转移的 DP，罐子分裂。

```
#define mod 10007
inline void insert(char *s){
    int p=0;
    for(int c,i=0;s[i];p=next[p][c],i++,d[p]=i){//需要记录节点深度的写法
        c=map[s[i]];
        if(!next[p][c])next[p][c]=newnode();
    }
    type[p]=1;
}
char s[100];
char str[20];
int f[2][N][110],len;
bool match(char *s){
    int p=0;
    for(int c;*s;s++){
        c=map[*s];
```

```
17          p=next[p][c];
18          if(type[p])return 0;
19      }
20      f[0][p][len]=1;
21      return 1;
22  }
23  void add(int &a,int b){
24      a+=b;
25      if(a>=mod)a-=mod;
26  }
27  int main(){
28      for(int i=0;i<D;i++)map[i+'a']=i;
29      ID=0;
30      newnode();
31      d[0]=0;
32      int p,n;
33      scanf("%s%d%d",s,&p,&n);
34      len=strlen(s);
35      while(n--){
36          scanf("%s",str);
37          insert(str);
38      }
39      bfs();
40      memset(f,0,sizeof(f));
41      if(match(s)==0){
42          puts("0 1");
43          return 0;
44      }
45      int now=0,a=0,b=0;
46      for(int i=0;i<p;now=!now,i++){//days
47          for(int j=0;j<ID;j++){//sta
48              if(type[ID])continue;
49              for(int k=min(100,len+p);k>0;k--){//len
50                  int *inc=&f[now][j][k];
51                  if(*inc==0)continue;
52                  if(k-1<d[j]) add(f[!now][fail[j]][k-1],*inc);
53                  else add(f[!now][j][k-1],*inc);
54                  for(int r=0;r<D;r++)//next
55                      if(type[next[j][r]])add(b,*inc);
56                      else add(f[!now][next[j][r]][k+1],*inc);
57                  *inc=0;
58              }
59          }
60          add(a,f[!now][0][0]);
61          f[!now][0][0]=0;
62      }
63      printf("%d %d\n",a,b);
64      return 0;
65  }
```

1.4 后缀数组

1.4.1 倍增算法

rk 从 0 开始
sa 从 1 开始, 因为最后一个字符 (最小的) 排在第 0 位
hi 从 2 开始, 因为表示的是 sa[i-1] 和 sa[i]
sa[i] 排第 i 位的后缀的下标
rk[i] 下标为 i 的后缀排第几

```
1 #define N 200010
2 int dp[20][N],R[32];
3 struct RMQ {//下标由开始注意0,和minmax
4     RMQ(){for(int i=0;R[i]=1<<i,R[i]<N;i++);}
5     void RM(int *a,int n) {
6         for(int i=0;i<n;i++)dp[0][i]=a[i];
```

```
        for(int i=1;R[i]<=n;i++)
            for(int j=0;j+R[i]<=n;j++)
                dp[i][j]=min(dp[i-1][j],dp[i-1][j+R[i-1]]);
    }
    int Q(int s,int e){
        int i=0,h=(e-s+1)>>1;
        while(R[i]<=h)i++;
        return min(dp[i][e-R[i]+1],dp[i][s]);
    }
} rmq;
#define inc(i,n) for(i=0;i<n;i++)
#define dec(i,n) for(i=n-1;i>=0;i--)
int bu[N],X[N],Y[N],sa[N],rk[N],hi[N],r[N];
bool cmp(int *r,int a,int b,int l){return r[a]==r[b]&&r[a+l]==r[b+l];}
#define busort(t)\
inc(i,m)bu[i]=0;\
inc(i,n)bu[x[t]]++;\
inc(i,m)bu[i+1]+=bu[i];\
dec(i,n)sa[--bu[x[t]]]=t;
//r[n-1]=0,suffix(r,n,200)
void suffix(int *r,int n,int m){
    int i,j,p,*x=X,*y=Y,*t;
    inc(i,n)x[i]=r[i];
    busort(i);
    for(j=1,p=1;p<n;m=p,j<<=1){
        inc(p,j)y[p]=n-j+p;//bug
        inc(i,n)if(sa[i]>=j)y[p++]=sa[i]-j;
        busort(y[i]);
        for(t=x,x=y,y=t,x[sa[0]]=0,i=1,p=1;i<n;i++)
            x[sa[i]]=cmp(y,sa[i-1],sa[i],j)?p-1:p++;
    }
    inc(i,n)rk[sa[i]]=i;//calhigh()
    for(int i=0,k=0;i<n-1;hi[rk[i++]]=k)
        for(k?k--:0,j=sa[rk[i]-1];r[i+k]==r[j+k];k++);
    rmq.RM(hi,n);
}
int lcp(int a,int b){
    a=rk[a],b=rk[b];
    if(a>b)swap(a,b);
    return rmq.Q(++a,b);
}
/subsection版本二，处理字符集很大的情况{}
bool sp(int a,int b){rt r[a]<r[b];}
void SA(int *r,int n){
    int m,i,j,p,*x=X,*y=Y,*t;
    inc(i,n)x[i]=r[i],sa[i]=i;
    sort(sa,sa+n,sp);
    for(j=1;;j<<=1){
        for(t=x,x=y,y=t,x[sa[0]]=0,i=1,p=1;i<n;i++)
            x[sa[i]]=cmp(y,sa[i-1],sa[i],j>>1)?p-1:p++;
        m=p;if(p>=n)break;
        inc(p,j)y[p]=n-j+p;
        inc(i,n)if(sa[i]>=j)y[p++]=sa[i]-j;
        inc(i,m)bu[i]=0;
        inc(i,n)bu[x[y[i]]]++;
        inc(i,m)bu[i+1]+=bu[i];
        dec(i,n)sa[--bu[x[y[i]]]]=y[i];
    }
}
```

1.4.2 不同回文子串的个数

```
#define ff(i,n) for(int i=0;i<n;i++)
char s[N];
int main(){
    int T,cas=1;
```

```
5       scanf("%d",&T);
6       while(T--){
7           scanf("%s",s);
8           int n=strlen(s);
9           ff(i,n)r[i]=s[i];
10          r[n]=127;
11          ff(i,n)r[1+i+n]=r[n-i-1];
12          n=n*2+2;
13          r[n-1]=0;
14          suffix(r,n,128);
15          int da[2]={},cnt=0;
16          ff(i,n)ff(j,2){
17              int res=lcp(sa[i],n-sa[i]-1-j);//最长公共前缀
18              if(da[j]>hi[i])da[j]=hi[i];
19              cnt+=max(0,res-da[j]);
20              if(da[j]<res)da[j]=res;
21          }
22          printf("Case #%d: %d\n",cas++,cnt);
23      }
24      return 0;
25  }
```

1.4.3 不同子矩阵的个数

不是整数类型，无法用桶放，第一次的排序所用快排

```
1  #include <stdio.h>
2  #include <algorithm>
3  using namespace std;
4  #define N 130
5  #define N2 N*N
6  #define rt return
7  char s[N];
8  int a[N][N];
9  const long long p=117,mod=1000000007;
10 #define inc(i,n) for(i=0;i<n;i++)
11 #define dec(i,n) for(i=n-1;i>=0;i--)
12 typedef pair<int,int> Hash;
13 Hash r[N2];
14 int bu[N2],X[N2],Y[N2],sa[N2],rk[N2],hi[N2];
15 bool cmp(int *r,int a,int b,int l){rt r[a]==r[b]&&r[a+l]==r[b+l];}
16 bool sp(int a,int b){rt r[a]<r[b];}
17 void SA(int n){
18     int m,i,j,p,*x=X,*y=Y,*t;
19     inc(i,n)sa[i]=i;
20     sort(sa,sa+n,sp);
21     for(i=1,p=1,x[sa[0]]=0;i<n;i++)
22         x[sa[i]]=(r[sa[i-1]]==r[sa[i]]?p-1:p++);
23     for(j=1,m=p,p=1;p<n;m=p,j<<=1){
24         inc(p,j)y[p]=n-j+p;
25         inc(i,n)if(sa[i]>=j)y[p++]=sa[i]-j;
26         inc(i,m)bu[i]=0;
27         inc(i,n)bu[x[y[i]]]++;
28         inc(i,m)bu[i+1]+=bu[i];
29         dec(i,n)sa[--bu[x[y[i]]]]=y[i];
30         for(t=x,x=y,y=t,x[sa[0]]=0,i=1,p=1;i<n;i++)
31             x[sa[i]]=cmp(y,sa[i-1],sa[i],j)?p-1:p++;
32     }
33     inc(i,n)rk[sa[i]]=i;
34     for(int i=0,k=0;i<n-1;hi[rk[i++]]=k)
35         for(k?k--:0,j=sa[rk[i]-1];r[i+k]==r[j+k];k++);
36 }
37 int main(){
38     int T,n,m,cas=1;
39     scanf("%d",&T);
40     while(T--){
41         scanf("%d%d",&n,&m);
42         for(int i=0;i<n;i++){
43             scanf("%s",s);
```

```
44              for(int j=0;j<m;j++)a[i][j]=s[j]-'A'+1;
45          }
46          int cnt=0;
47          for(int h=0;h<n;h++){
48              int sz=0;
49              for(int i=0;i+h<n;i++,sz++){
50                  for(int j=0;j<m;j++,sz++){
51                      if(h==0)r[sz]=make_pair(a[i+h][j],a[i+h][j]);
52                      else r[sz]=make_pair((p*r[sz].first+a[i+h][j])%mod,r[sz].
                            second+a[i+h][j]);
53                  }
54                  if(h==0)r[sz]=make_pair(mod+i,mod+i);
55              }
56              r[sz-1]=make_pair(0,0);
57              SA(sz);
58              cnt+=(1+m)*m/2*(n-h);
59              for(int i=2;i<sz;i++)cnt-=hi[i];
60          }
61          printf("Case #%d: %d\n",cas++,cnt);
62      }
63  }
```

1.4.4 DC3 算法

```
 1  #define F(x) ((x)/3+((x)%3==1?0:tb))
 2  #define G(x) ((x)<tb?(x)*3+1:((x)-tb)*3+2)
 3  int wa[maxn],wb[maxn],wv[maxn],ws[maxn];
 4  int c0(int *r,int a,int b)
 5  {return r[a]==r[b]&&r[a+1]==r[b+1]&&r[a+2]==r[b+2];}
 6  int c12(int k,int *r,int a,int b)
 7  {if(k==2) return r[a]<r[b]||r[a]==r[b]&&c12(1,r,a+1,b+1);
 8   else return r[a]<r[b]||r[a]==r[b]&&wv[a+1]<wv[b+1];}
 9  void sort(int *r,int *a,int *b,int n,int m)
10  {
11      int i;
12      for(i=0;i<n;i++) wv[i]=r[a[i]];
13      for(i=0;i<m;i++) ws[i]=0;
14      for(i=0;i<n;i++) ws[wv[i]]++;
15      for(i=1;i<m;i++) ws[i]+=ws[i-1];
16      for(i=n-1;i>=0;i--) b[--ws[wv[i]]]=a[i];
17      return;
18  }
19  void dc3(int *r,int *sa,int n,int m)
20  {
21      int i,j,*rn=r+n,*san=sa+n,ta=0,tb=(n+1)/3,tbc=0,p;
22      r[n]=r[n+1]=0;
23      for(i=0;i<n;i++) if(i%3!=0) wa[tbc++]=i;
24      sort(r+2,wa,wb,tbc,m);
25      sort(r+1,wb,wa,tbc,m);
26      sort(r,wa,wb,tbc,m);
27      for(p=1,rn[F(wb[0])]=0,i=1;i<tbc;i++)
28      rn[F(wb[i])]=c0(r,wb[i-1],wb[i])?p-1:p++;
29      if(p<tbc) dc3(rn,san,tbc,p);
30      else for(i=0;i<tbc;i++) san[rn[i]]=i;
31      for(i=0;i<tbc;i++) if(san[i]<tb) wb[ta++]=san[i]*3;
32      if(n%3==1) wb[ta++]=n-1;
33      sort(r,wb,wa,ta,m);
34      for(i=0;i<tbc;i++) wv[wb[i]=G(san[i])]=i;
35      for(i=0,j=0,p=0;i<ta && j<tbc;p++)
36      sa[p]=c12(wb[j]%3,r,wa[i],wb[j])?wa[i++]:wb[j++];
37      for(;i<ta;p++) sa[p]=wa[i++];
38      for(;j<tbc;p++) sa[p]=wb[j++];
39      return;
40  }
```

1.5 Dancing links

resume 和 remove 容易敲错遍历顺序很有影响，必要时候可以更改顺序可以对列计数器做特判，以减少时间

```
#define N -1
#define D -1
int l[N],r[N],u[N],d[N],x[N],y[N],c[N],id,col;
#define ff(i,d,t) for(int i=d[t];i!=t;i=d[i])
void link(int i,int j,int &k){
    c[j]++;
        x[id]=i;//行标保存决策，非必要,
    y[id]=d[id]=j;
    u[id]=u[j];
    u[j]=d[u[j]]=id;
    r[id]=id-k;
    l[id-k]=id;
    if(k){
        l[id]=id-1;
        r[id-1]=id;
    }
    k++;
    id++;
}
void init(int m){
    col=m;
    for(int i=0;i<=col;i++){
        l[i]=i-1;
        r[i]=i+1;
        u[i]=d[i]=y[i]=i;
        c[i]=0;
    }
    l[0]=col;
    r[col]=0;
    id=col+1;
        best=10000;//重复覆盖需要初始化
}
```

1.5.1 精确覆盖

```
inline void remove(int t){
    ff(i,d,t){
        if(i>col)ff(j,r,i)
        u[d[j]]=u[j],d[u[j]]=d[j],c[y[j]]--;
        l[r[i]]=l[i],r[l[i]]=r[i];
    }
}
inline void resume(int t){
    ff(i,u,t){
        l[r[i]]=r[l[i]]=i;
        if(i>col)ff(j,l,i)
        u[d[j]]=d[u[j]]=j,++c[y[j]];
    }
}
bool dfs(int k){
    if(r[0]==0){cnt=k;return 1;}
    int t=r[0];
    ff(i,r,0)if(c[i]<c[t])t=i;
    if(c[t]==0)return 0;
    ff(i,d,t){
        remove(i);
        ff(j,r,i)remove(j);
        o[k]=x[i];
        bool f=dfs(k+1);
        ff(j,l,i)resume(j);
        resume(i);
        if(f)return 1;
```

```
28      }
29      return 0;
30  }
```

1.5.2 重复覆盖

```
 1  inline void remove(int t){
 2      ff(i,d,t)l[r[i]]=l[i],r[l[i]]=r[i];
 3  }
 4  inline void resume(int t){
 5      ff(i,u,t)l[r[i]]=r[l[i]]=i;
 6  }
 7  int astar(){
 8      int cnt=0;
 9      bool h[D]={};//列的数目*********
10      ff(t,r,0)if(!h[t]){
11          h[t]=1,cnt++;
12          ff(i,d,t)ff(j,r,i)h[y[j]]=1;
13      }
14      return cnt;
15  }
16  void dfs(int k){
17      if(k+astar()>=best)return;
18      if(r[0]==0){best=k;return;}
19      int t=r[0];
20      ff(i,r,0)if(c[i]<c[t])t=i;
21          if(c[t]==0)return;
22      ff(i,d,t){
23          remove(i);
24          ff(j,r,i)remove(j);
25          dfs(k+1);
26          ff(j,l,i)resume(j);
27          resume(i);
28      }
29  }
```

1.5.3 选取最少的行使得某些列覆盖至少一次某些列至多一次

sz[n] 以前的列至少一次

```
 1  int astar(){
 2      int cnt=0;
 3      bool h[55]={};
 4      for(int t=r[0];t<sz[n];t=r[t])//
 5          if(!h[t]){
 6              h[t]=1,cnt++;
 7              ff(i,d,t)ff(j,r,i)h[y[j]]=1;
 8          }
 9      return cnt;
10  }
11  void dfs(int k){
12      if(k+astar()>=best)return;
13      if(r[0]>=sz[n]||r[0]==0){best=k;return;}
14      int t=r[0];
15      for(int i=r[0];i<sz[n];i=r[i])
16          if(c[i]<c[t])t=i;
17      if(c[t]==0)return;
18      ff(i,d,t){
19          remove(i);
20          ff(j,r,i)if(y[j]>=sz[n])exremove(j);
21          else remove(j);
22          dfs(k+1);
23          ff(j,l,i)if(y[j]>=sz[n])exresume(j);
24          else resume(j);
25          resume(i);
26      }
27  }
```

1.5.4 数独问题

```
#define N 1000000
char s[20][20];
int cnt,o[300];
int D=16,dd=4;
int main(){
    while(~scanf("%s",s[0])){
        for(int i=1;i<D;i++)scanf("%s",s[i]);
        init(D*D*4);
        int row=0;
        for(int k=0;k<D;k++){
            int t=k*D*3+1;
            for(int i=0;i<D;i++)
                for(int j=0,cnt=0;j<D;j++,row++){
                    if(s[i][j]!='-'&&s[i][j]!='A'+k)continue;
                    link(row,t+i,cnt);
                    link(row,t+D+j,cnt);
                    link(row,t+D+D+(i/dd)*dd+j/dd,cnt);
                    link(row,1+D*D*3+i*D+j,cnt);
                }
        }
        dfs(0);
        for(int i=0;i<cnt;i++)
            s[o[i]/D%D][o[i]%D]=o[i]/(D*D)+'A';
        for(int i=0;i<D;i++)
            puts(s[i]);
        puts("");
    }
    return 0;
}
```

1.6 Splay tree

具体题目修改 push down,push up,newnode,op,make 等

1.6.1 替代常规线段树

pushdown 的时候注意更新所有 pushup 需要的变量

```
#include <stdio.h>
#define N 10010
#define key (ch[ch[root][1]][0])
#define oo 2000000000
typedef long long LL;
int need[20];
int mi[N],exp[N],lev[N],add[N],mxl[N],mx[N];
int ch[N][2],pre[N],sz[N];//固有变量
int root,id;
#define min(a,b) ((a)<(b)?(a):(b))
inline void update_mx(int &x,int &id,int xx,int iid){
    if(x<xx)x=xx,id=iid;
}
inline void update_exp(int &x,int &k,int c){
    x+=k*c;
    while(x>=need[k+1])k++;
}
void push_down(int x){
    if(add[x]){
        update_exp(exp[x],lev[x],add[x]);
        update_exp(mx[x],mxl[x],add[x]);
        mi[x]-=add[x];//
        int l=ch[x][0],r=ch[x][1];
        if(l)add[l]+=add[x];
        if(r)add[r]+=add[x];
```

```
26          add[x]=0;//
27      }
28  }
29  void push_up(int x){
30      int l=ch[x][0],r=ch[x][1];
31      push_down(x);
32      if(l){
33          if(add[l]>=mi[l])push_up(l);//递归更新，特殊～～
34          else push_down(l);
35      }
36      if(r){
37          if(add[r]>=mi[r])push_up(r);
38          else push_down(r);
39      }
40      sz[x]=1+sz[l]+sz[r];
41      mx[x]=exp[x],mxl[x]=lev[x];//的某些值不能滞留splay
42      update_mx(mx[x],mxl[x],mx[l],mxl[l]);
43      update_mx(mx[x],mxl[x],mx[r],mxl[r]);
44      mi[x]=(need[lev[x]+1]-exp[x]-1)/lev[x]+1;//去滞留
45      mi[x]=min(mi[x],mi[l]);
46      mi[x]=min(mi[x],mi[r]);
47  }
48  int newnode(int f){
49      int x=id++;//这里包括空节点的信息
50      mi[x]=oo,mxl[x]=lev[x]=1,exp[x]=add[x]=mx[x]=0;
51      ch[x][0]=ch[x][1]=0;
52      sz[x]=1;
53      pre[x]=f;
54      return x;
55  }
56  int make(int l,int r,int f){
57      if(l>r)return 0;
58      int m=(l+r)>>1,x=newnode(f);
59      ch[x][0]=make(l,m-1,x);
60      ch[x][1]=make(m+1,r,x);
61      push_up(x);
62      return x;
63  }
64  void rot(int x,int f){
65      int y=pre[x],z=pre[y];
66      push_down(y);
67      push_down(x);
68      ch[y][!f]=ch[x][f];
69      pre[ch[x][f]]=y;
70      pre[x]=z;
71      if(z)ch[z][ch[z][1]==y]=x;
72      ch[x][f]=y;
73      pre[y]=x;
74      push_up(y);
75  }
76  void splay(int x,int goal){
77      push_down(x);
78      while(pre[x]!=goal){
79          int y=pre[x],z=pre[y];
80          if(z==goal){
81              rot(x,ch[y][0]==x);
82          }else{
83              int f=(ch[z][0]==y);
84              (ch[y][f]==x)?rot(x,!f):rot(y,f);
85              rot(x,f);
86          }
87      }
88      push_up(x);
89      if(goal==0)root=x;
90  }
91  void init(int n){
92      id=0;
```

```
 93      newnode(0);
 94      sz[0]=0;
 95      root=newnode(0);
 96      ch[root][1]=newnode(root);
 97      key=make(0,n-1,ch[root][1]);
 98      splay(key,0);
 99  }
100  int getk(int k,int x){
101      push_down(x);
102      if(sz[ch[x][0]]==k)return x;
103      if(sz[ch[x][0]]>k)return getk(k,ch[x][0]);
104      return getk(k-sz[ch[x][0]]-1,ch[x][1]);
105  }
106  void rotk(int k,int goal){
107      splay(getk(k,root),goal);
108  }
109  char s[2];
110  void op(){
111      int l,r,e;
112      scanf("%s%d%d",s,&l,&r);
113      rotk(l-1,0);
114      rotk(r+1,root);
115      if(s[0]=='W'){
116          scanf("%d",&e);
117          add[key]+=e;
118      }else printf("%d\n",mx[key]);
119  }
120  int main(){
121      int T,n,m,q,cas=1;
122      scanf("%d",&T);
123      while(T--){
124          scanf("%d%d%d",&n,&m,&q);
125          need[1]=0,need[m+1]=oo;
126          for(int i=2;i<=m;i++)scanf("%d",&need[i]);
127          init(n);
128          printf("Case␣%d:\n",cas++);
129          while(q--)op();
130          puts("");
131      }
132      return 0;
133  }
```

1.6.2 插入删除修改翻转求和求最大子列

erase: 删除包括 x 及其子节点在内的所有节点，注意孩子更为 0

```
 1  #include <stdio.h>
 2  #include <limits.h>
 3  #include <algorithm>
 4  #include <string.h>
 5  using namespace std;
 6  #define N 600010
 7  #define inf 100000000
 8  #define key (ch[ch[root][1]][0])
 9
10  int ch[N][2];
11  int pre[N];
12  int sz[N];
13  int val[N];
14  int top1, top2, root;
15  int ss[N], q[N];
16
17  bool same[N];
18  bool rev[N];
19  int sum[N];
20  int ma[N], ml[N], mr[N];
```

```
21
22  void push_down(int x) {
23      int l = ch[x][0], r = ch[x][1];
24      if (same[x]) {
25          same[x] = 0;
26          same[l] = 1;
27          same[r] = 1;
28          val[l] = val[r] = val[x];
29          sum[x] = val[x] * sz[x];
30          ma[x] = ml[x] = mr[x] = max(val[x], sum[x]);
31      }
32      if (rev[x]) {
33          rev[x] = 0;
34          rev[l] = !rev[l];
35          rev[r] = !rev[r];
36          swap(ch[x][0], ch[x][1]);
37          swap(ml[x], mr[x]);
38      }
39  }
40
41  void push_up(int x) {
42      int l = ch[x][0], r = ch[x][1];
43      push_down(x);
44      if(l)push_down(l);
45      if(r)push_down(r);
46      sz[x] = 1 + sz[l] + sz[r];
47      sum[x] = val[x] + sum[l] + sum[r];
48      ml[x]=max(ml[l],sum[l]+val[x]+max(0,ml[r]));
49      mr[x]=max(mr[r],sum[r]+val[x]+max(0,mr[l]));
50      ma[x]=max(ma[l],ma[r]);
51      ma[x]=max(ma[x],max(ml[x],mr[x]));
52      ma[x]=max(ma[x],val[x]+max(0,mr[l])+max(0,ml[r]));
53  }
54
55  void erase(int x) {
56      ss[top2++] = x;
57      if (ch[x][0])erase(ch[x][0]);
58      if (ch[x][1])erase(ch[x][1]);
59  }
60
61  int newnode(int f) {
62      int x;
63      if (top2)x = ss[--top2];
64      else x = ++top1;
65      ch[x][0] = ch[x][1] = rev[x] = same[x] = 0;
66      sz[x] = 1;
67      pre[x] = f;
68      return x;
69  }
70
71  int make(int l, int r, int f) {//新建的子树f
72      if (l > r)return 0;
73      int m = (l + r) >> 1;
74      int x = newnode(f);
75      ch[x][0] = make(l, m - 1, x);
76      scanf("%d", &val[x]);
77      ch[x][1] = make(m + 1, r, x);
78      push_up(x);
79      return x;
80  }
81
82  void rot(int x, int f) {
83      int y = pre[x], z = pre[y];
84      push_down(y);
85      push_down(x);
86      ch[y][!f] = ch[x][f];
87      pre[ch[x][f]] = y;
```

```
88      pre[x] = z;
89      if (z)ch[z][ch[z][1] == y] = x;
90      ch[x][f] = y;
91      pre[y] = x;
92      push_up(y);
93  }
94
95  void splay(int x, int goal) {
96      push_down(x);
97      while (pre[x] != goal) {
98          int y = pre[x], z = pre[y];
99          if (z == goal) {
100             rot(x, ch[y][0] == x);
101         } else {
102             int f = (ch[z][0] == y);
103             (ch[y][f] == x) ? rot(x, !f) : rot(y, f);
104             rot(x, f);
105         }
106     }
107     push_up(x);
108     if (goal == 0)root = x;
109 }
110
111 void init(int n) {//初始化
112     val[0] = ma[0] = ml[0] = mr[0] = -inf;//空节点信息很重要
113     sum[0] = sz[0]= 0;
114     top1 = top2 = 0;
115     root = newnode(0);
116     val[root] = -inf;//bug
117     ch[root][1] = newnode(root);
118     val[ch[root][1]] = -inf; //bug
119     key = make(0, n - 1, ch[root][1]);
120     splay(key, 0);
121 }
122
123 int getk(int k, int x) {
124     push_down(x);
125     if (sz[ch[x][0]] == k)return x;
126     if (sz[ch[x][0]] > k)return getk(k, ch[x][0]);
127     return getk(k - sz[ch[x][0]] - 1, ch[x][1]);
128 }
129
130 void rotk(int k, int goal) {
131     splay(getk(k, root), goal);
132 }
133
134 void insert() {
135     int x, n;
136     scanf("%d%d", &x, &n);
137     rotk(x, 0);
138     rotk(x + 1, root);
139     key = make(0, n - 1, ch[root][1]);
140     splay(key, 0);
141 }
142
143 void del() {
144     int x, k;
145     scanf("%d%d", &x, &k);
146     rotk(x - 1, 0);
147     rotk(x + k, root);
148     erase(key);
149     key = 0;
150     splay(ch[root][1], 0);
151 }
152
153 void make_same() {
154     int x, y, z;
```

```
155     scanf("%d%d%d", &x, &y, &z);
156     rotk(x - 1, 0);
157     rotk(x + y, root);
158     x = key;
159     same[x] = 1;
160     val[x] = z;
161     splay(key, 0);
162 }
163
164 void reverse() {
165     int x, k;
166     scanf("%d%d", &x, &k);
167     rotk(x - 1, 0);
168     rotk(x + k, root);
169     rev[key] = !rev[key];
170     splay(key, 0);
171 }
172
173 void get_sum() {//区间和
174     int x, k;
175     scanf("%d%d", &x, &k);
176     rotk(x - 1, 0);
177     rotk(x + k, root);
178     printf("%d\n", sum[key]);
179 }
180
181 void max_sum() {//最大子段和
182     push_down(root);
183     push_up(root);
184     printf("%d\n", ma[root]);
185 }
186
187 void debug(int x) {
188     if (x == 0)return;
189     push_down(x);
190     debug(ch[x][0]);
191     printf("id=%2d␣l=%2d␣r=%2d␣sz=%2d␣val=%2d␣sum=%2d␣ml=%2d␣mr=%2d␣ma=%2d\n"
          ,
192                    x, ch[x][0], ch[x][1], sz[x], val[x], sum[x], ml[x], mr[x
                       ], ma[x]);
193     debug(ch[x][1]);
194 }
195
196 void debug() {
197     printf("root=%d\n", root);
198     debug(root);
199 }
200
201 int main() {
202     int n, m;
203     scanf("%d%d", &n, &m);
204     init(n);
205     //st.debug();
206     while (m--) {
207         char s[100];
208         scanf("%s", s);
209         if (strcmp(s, "INSERT") == 0) {
210             insert();
211         } else if (strcmp(s, "DELETE") == 0) {
212             del();
213         } else if (strcmp(s, "MAKE-SAME") == 0) {
214             make_same();
215         } else if (strcmp(s, "REVERSE") == 0) {
216             reverse();
217         } else if (strcmp(s, "GET-SUM") == 0) {
218             get_sum();
219         } else if (strcmp(s, "MAX-SUM") == 0) {
```

```
220                 max_sum();
221             }
222             //debug();
223         }
224 }
```

1.7 搜索

1.7.1 8 数码双向广搜

记得初始化

```
  1 int fac[10],ans=-1,v[362880]={0},dir[4]={-3,-1,1,3},mulfac[9][9];
  2 bool forbid[9][4]={
  3             {1,1,0,0},
  4             {1,0,0,0},
  5             {1,0,1,0},
  6             {0,1,0,0},
  7             {0,0,0,0},
  8             {0,0,1,0},
  9             {0,1,0,1},
 10             {0,0,0,1},
 11             {0,0,1,1}
 12 };
 13 struct point{
 14     int a[9],id,x;
 15     void get(){
 16         for(int i=0,t;i<9;i++){
 17             scanf("%d",&t);
 18             if(t==0)x=i;
 19             a[i]=t;
 20         }
 21         map();
 22     }
 23     void swap(int y){
 24         int t=a[y];
 25         a[y]=a[x];
 26         a[x]=t;
 27         x=y;
 28     }
 29     void map(){
 30         id=0;
 31         for(int i=1;i<9;i++){
 32             int c=0;
 33             for(int j=0;j<i;j++)
 34                 if(a[j]>a[i])c++;
 35             id+=mulfac[c][i];
 36         }
 37     }
 38     int res(){
 39         int cnt=0;
 40         for(int i=0;i<9;i++){
 41             if(a[i]==0)continue;
 42             for(int j=0;j<i;j++)
 43                 if(a[j]>a[i])cnt++;}
 44         return cnt&1;
 45     }
 46 };
 47 struct queue{
 48     int fg,step;
 49     point q[10000],*s,*e;
 50     int init(int x){
 51         s=q;
 52         e=q+1;
 53         fg=x;
 54         s->get();
```

```
55          v[s->id]=fg;
56          step=0;
57          return s->res();
58      }
59      int promote(){
60          for(point *t=e;s!=t;s++){
61              for(int i=0;i<4;i++){
62                  if(forbid[s->x][i])continue;
63                  point p=*s;
64                  p.swap(p.x+dir[i]);
65                  p.map();
66                  if(!v[p.id]){
67                      v[p.id]=fg;
68                      *e++=p;
69                  }else if(v[p.id]!=fg)return 0;
70              }
71          }
72          step++;
73          return 1;
74      }
75  }q1,q2;
76  int main(){
77      fac[0]=1;
78      for(int i=1;i<9;i++)
79          fac[i]=i*fac[i-1];
80      for(int i=0;i<9;i++){
81          for(int j=0;j<9;j++)
82              mulfac[i][j]=i*fac[j];
83      }
84      if(q1.init(1)==q2.init(2)){
85          if(q1.s->id==q2.s->id)ans=0;
86          else{
87              while(q1.promote()&&q2.promote());
88              ans=q1.step+q2.step+1;
89          }
90      }
91      printf("%d\n",ans);
92      return 0;
93  }
```

1.7.2 15 数码 IDA*

```
 1  int a[16], k, p;
 2  char path[100];
 3  char ch[] = {'L', 'U', 'D', 'R'};
 4  int dir[] = {-1, -4, 4, 1};
 5  int m[16][16] = {0};
 6  bool go[][4] = {
 7      0, 0, 1, 1,
 8      1, 0, 1, 1,
 9      1, 0, 1, 1,
10      1, 0, 1, 0,
11      0, 1, 1, 1,
12      1, 1, 1, 1,
13      1, 1, 1, 1,
14      1, 1, 1, 0,
15      0, 1, 1, 1,
16      1, 1, 1, 1,
17      1, 1, 1, 1,
18      1, 1, 1, 0,
19      0, 1, 0, 1,
20      1, 1, 0, 1,
21      1, 1, 0, 1,
22      1, 1, 0, 0,
23  };
24
```

```
25 struct point {
26     int x, y;
27
28     bool operator<(const point & p)const {
29         return x < p.x;
30     }
31 };
32
33 void swap(int i) {
34     a[p] = a[p + dir[i]];
35     a[p += dir[i]] = 0;
36 }
37
38 bool dfs(int step, int mht, int x) {
39     if (mht == 0) {
40         path[step] = 0;
41         puts(path);
42         return 1;
43     }
44     int cnt = 0;
45     point c[4];
46     for (int i = 0; i < 4; i++) {
47         if (!go[p][i] || i == x)continue;
48         int t = p + dir[i];
49         c[cnt].x = mht + m[p][a[t] - 1] - m[t][a[t] - 1];
50         if (c[cnt].x + step <= k)c[cnt++].y = i;
51     }
52     sort(c, c + cnt);
53     for (int i = 0; i < cnt; i++) {
54         path[step] = ch[c[i].y];
55         swap(c[i].y);
56         if (dfs(step + 1, c[i].x, 3 - c[i].y))return 1;
57         swap(3 - c[i].y);
58     }
59     return 0;
60 }
61
62 int main() {
63     for (int i = 0; i < 16; i++)
64         for (int j = i + 1; j < 16; j++)
65             m[i][j] = m[j][i] = abs(i / 4 - j / 4) + abs(i % 4 - j % 4);
66     int T;
67     scanf("%d", &T);
68     while (T--) {
69         int cnt = 0;
70         for (int i = 0; i < 16; i++) {
71             scanf("%d", &a[i]);
72             if (a[i] == 0)p = i;
73             for (int j = 0; j < i; j++)
74                 cnt += a[j] > a[i];
75         }
76         if (((cnt + m[p][15])&1) == 0)
77             puts("This puzzle is not solvable.");
78         else {
79             int mht = 0;
80             for (int i = 0; i < 16; i++)
81                 if (a[i]) mht += m[i][a[i] - 1];
82             for (k = mht; k < 51 && !dfs(0, mht, -1); k++);
83         }
84     }
85     return 0;
86 }
```

1.7.3 使用回滚函数

```
1 int cnt[500];
2 int a[500];
3 vector<int> ans[10000];
```

```
vector<int> res;
int id, l, r, suml, sumr;
int m, n;

void rollback(int pos, bool type) {
    if (type == 0) {
        int x = 0;
        for (int i = l; i > pos; i--) {
            x += res[i];
            cnt[x]++;
        }
    } else {
        int x = 0;
        for (int i = l; i >= 0; i--) {
            x += res[i];
            cnt[x]++;
        }
        x = 0;
        for (int i = r; i < pos; i++) {
            x += res[i];
            cnt[x]++;
        }
    }
}

bool check(int x, bool type) {
    if (type == 0) {
        if (x <= sumr)return 0;
        res[r] = x - sumr;
        res[l] = a[m - 1] - x - suml;
        if (res[l] <= 0)return 0;
    } else {
        if (x <= suml)return 0;
        res[l] = x - suml;
        res[r] = a[m - 1] - x - sumr;
        if (res[r] <= 0)return 0;
    }
    x = 0;
    for (int i = l; i >= 0; i--) {
        x += res[i];
        if (cnt[x] == 0) {
            rollback(i, 0);
            return 0;
        }
        cnt[x]--;
    }
    x = 0;
    for (int i = r; i < n; i++) {
        x += res[i];
        if (cnt[x] == 0) {
            rollback(i, 1);
            return 0;
        }
        cnt[x]--;
    }
    return 1;
}

void dfs(int lim) {
    if (l == r) {
        ans[id++] = res;
        return;
    }
    while (cnt[a[lim]] == 0)lim--;
    if (check(a[lim], 0)) {
        suml += res[l];
        l++;
```

```
71          dfs(lim);
72          l--;
73          suml -= res[l];
74          res[r] = a[lim] - sumr;
75          rollback(n, 1);
76      }
77      if (check(a[lim], 1)) {
78          sumr += res[r];
79          r--;
80          dfs(lim);
81          r++;
82          sumr -= res[r];
83          res[l] = a[lim] - suml;
84          rollback(n, 1);
85      }
86  }
87
88  int main() {
89      while (scanf("%d", &n), n) {
90          memset(cnt, 0, sizeof (cnt));
91          m = n * (n - 1) / 2;
92          n--;
93          for (int i = 0; i < m; i++) {
94              scanf("%d", &a[i]);
95              cnt[a[i]]++;
96          }
97          sort(a, a + m);
98          m = unique(a, a + m) - a;
99          id = 0;
100         res.resize(n);
101         l = 0, r = n - 1, suml = 0, sumr = 0;
102         cnt[a[m-1]]--;
103         res[0]=a[m-1];
104         dfs(m - 1);
105         sort(ans, ans + id);
106         id = unique(ans, ans + id) - ans;
107         for (int i = 0; i < id; i++) {
108             for (int j = 0; j < n; j++) {
109                 printf(j == n - 1 ? "%d\n" : "%d ", ans[i][j]);
110             }
111         }
112         puts("-----");
113     }
114     return 0;
115 }
```

1.8 线段树

1.8.1 矩形覆盖 k 层面积

```
1  #include <stdio.h>
2  #include <algorithm>
3  #include <iostream>
4  using namespace std;
5  #define N 30010
6  int e;
7  int hush[N * 2];
8  #define get_id(x) (lower_bound(hush+1,hush+n,x)-hush)
9
10 struct event {
11     int x, y1, y2, c;
12
13     event() {
14     }
15
16     event(int x, int y1, int y2, int c) : x(x), y1(y1), y2(y2), c(c) {
17     }
```

```

    bool operator<(const event & p)const {
        return x < p.x;
    }
} eve[N * 2];
int c[N * 6][11] = {};
int L, R, C, n, k;

void insert(int l, int r, int i) {
    int m = (l + r) >> 1, lc = i << 1, rc = lc + 1;
    if (L <= l && r <= R) {
        c[i][0] += C;
    } else {
        if (L < m)insert(l, m, lc);
        if (R > m)insert(m, r, rc);
    }
    for (int j = 1; j <= k; j++) {
        if (c[i][0] >= j)c[i][j] = hush[r] - hush[l];
        else if (l + 1 == r)c[i][j] = 0;
        else c[i][j] = c[lc][j - c[i][0]] + c[rc][j - c[i][0]];
    }
}

void insert(event & e) {
    L = get_id(e.y1), R = get_id(e.y2);
    C = e.c;
    insert(1, n, 1);
}

int main() {
    int T, cas = 1;
    scanf("%d", &T);
    while (T--) {
        n = 1;
        e = 0;
        int m;
        scanf("%d%d", &m, &k);
        for (int i = 0; i < m; i++) {
            int x1, y1, x2, y2;
            scanf("%d%d%d%d", &x1, &y1, &x2, &y2);
            x2++, y2++;
            hush[n++] = y1;
            hush[n++] = y2;
            eve[e++] = event(x1, y1, y2, 1);
            eve[e++] = event(x2, y1, y2, -1);
        }
        sort(hush + 1, hush + n);
        n = unique(hush + 1, hush + n) - hush;
        sort(eve, eve + e);
        long long ans = 0;
        for (int i = 0; i < e; i++) {
            insert(eve[i]);
            ans += 1ll * c[1][k]*(eve[i + 1].x - eve[i].x);
        }
        cout << "Case␣" << (cas++) << ":␣" << ans << endl;
    }
    return 0;
}
```

1.9 树状数组

1.9.1 树状数组上二分查找

记录数值为 i 的数的个数，利用树状数组查找第 k 大的数。

```
#define N 100010
#define max(a,b) (a>b?a:b)
```

```
3  int n,T,a[N],c[N],cas=1;
4  void update(int i,int x){
5      while(i<=n){
6          c[i]+=x;
7          i+=i&-i;
8      }
9  }
10 void add(int i,int x){
11     while(i<=n){
12         c[i]=max(c[i],x);
13         i+=i&-i;
14     }
15 }
16 int get(int i){
17     int ans=0;
18     while(i){
19        ans=max(c[i],ans);
20        i-=i&-i;
21     }
22     return ans;
23 }
24 int kth(int k){
25     int ret=0;
26     for(int i=17;i>=0;i--){//注意范围
27         ret+=(1<<i);
28         if(ret>n||c[ret]>=k)ret-=(1<<i);
29         else k-=c[ret];
30     }
31     return ret+1;
32 }
33 int main(){
34     scanf("%d",&T);
35     while(T--){
36         scanf("%d",&n);
37         for(int i=n;i>0;i--)c[i]=0;
38         for(int i=1;i<=n;i++){
39             scanf("%d",&a[i]);
40             update(i,1);
41         }
42         for(int i=n;i>0;i--){
43             a[i]=kth(a[i]+1);
44             update(a[i],-1);
45         }
46         printf("Case␣#%d:\n",cas++);
47         for(int i=n;i>0;i--)c[i]=0;
48         int ret=0;
49         for(int i=1;i<=n;i++){
50             int t=get(a[i])+1;
51             if(t>ret)ret=t;
52             add(a[i],t);
53             printf("%d\n",ret);
54         }
55         puts("");
56     }
57     return 0;
58 }
```

1.9.2 多维树状数组

```
1 #define N 110
2 using namespace std;
3 int c[N][N][N];
4 int n; //important
5 #define dec(i,x) for(int i=x;i>0;i-=i&-i)
6 #define inc(i,x) for(int i=x;i<=n;i+=i&-i)
7
```

```
 8 void update(int x, int y, int z) {
 9     dec(i, x)dec(j, y)dec(k, z)c[i][j][k] = !c[i][j][k];
10 }
11
12 int sum(int x, int y, int z) {
13     int s = 0;
14     inc(i, x)inc(j, y)inc(k, z)s ^= c[i][j][k];
15     return s;
16 }
17
18 int main() {
19     int m, op, x1, y1, z1, x2, y2, z2, ans;
20     while (~scanf("%d%d", &n, &m)) {
21         memset(c, 0, sizeof (c));
22         while (m--) {
23             scanf("%d%d%d%d", &op, &x1, &y1, &z1);
24             if (op == 1) {
25                 x1--, y1--, z1--;
26                 scanf("%d%d%d", &x2, &y2, &z2);
27                 update(x2, y2, z2);
28                 update(x2, y1, z2);
29                 update(x1, y2, z2);
30                 update(x2, y2, z1);
31
32                 update(x1, y1, z2);
33                 update(x2, y1, z1);
34                 update(x1, y2, z1);
35                 update(x1, y1, z1);
36             } else {
37                 ans = sum(x1, y1, z1);
38                 printf("%d\n", ans);
39             }
40         }
41     }
42     return 0;
43 }
```

1.9.3 区间更新，区间求和

```
 1 struct BIT{//[l,r]
 2     int n;
 3     LL a[N],b[N];
 4     BIT(int x=0){
 5         n=x+2;
 6         for(int i=n;i>0;i--)a[i]=b[i]=0;
 7     }
 8     void add1(int x,LL d){
 9         while(x<=n){a[x]+=d;x+=x&-x;}
10     }
11     LL cal1(int x){
12         LL s=0;
13         while(x){s+=a[x];x-=x&-x;}
14         return s;
15     }
16     int add2(int x,LL d){
17         while(x){b[x]+=d;x-=x&-x;}
18     }
19     LL cal2(int x){
20         LL s=0;
21         while(x<=n){s+=b[x];x+=x&-x;}
22         return s;
23     }
24     void add3(int x,LL d){
25         add1(x,d*x);add2(x-1,d);
26     }
27     LL cal3(int x){
28         return cal1(x)+cal2(x)*x;
29     }
```

```
30      void add(int l,int r,LL d){
31          l+=2;r+=2;
32          add3(l-1,-d);add3(r,d);
33      }
34      LL cal(int l,int r){
35          l+=2;r+=2;
36          return cal3(r)-cal3(l-1);
37      }
38  };// BIT bit(n);
```

1.10 并查集

按秩合并，一般不用 void Union(int a,int b) a=Find(a);b=Find(b); if(a==b) return; if(r[a]>r[b]) p[b]=a; else p[a]=b; if(r[a]==r[b]) r[b]++;

1.10.1 判奇圈，*A Bug's Life*

```
1  int p[3000];
2  bool r[3000];
3
4  int find(int x) {
5      if (x != p[x]) {
6          int t = p[x];
7          p[x] = find(p[x]);
8          r[x] = r[x]^r[t];
9      }
10     return p[x];
11 }
12 int main() {
13     int t, n, m, a, b, x, y;
14     scanf("%d", &t);
15     for (int i = 1; i <= t; i++) {
16         bool f = 0;
17         scanf("%d%d", &n, &m);
18         for (int j = 1; j <= n; j++)p[j] = j, r[j] = 0;
19         while (m--) {
20             scanf("%d%d", &a, &b);
21             if (!f) {
22                 x = find(a), y = find(b);
23                 if (x == y) {
24                     if (r[a] == r[b])f = 1;
25                 } else {
26                     p[x] = y;
27                     r[x] = !(r[a]^r[b]);
28                 }
29             }
30         }
31         printf("Scenario␣#%d:\n", i);
32         if (f)puts("Suspicious␣bugs␣found!\n");
33         else puts("No␣suspicious␣bugs␣found!\n");
34     }
35     return 0;
36 }
```

1.10.2 食物链

```
1  int p[50001],r[50001]={0};
2  int find(int x){
3          if(p[x]!=x){
4                  int t=p[x];
5                  p[x]=find(p[x]);
6                  r[x]=(r[x]+r[t])%3;
7          }
8          return p[x];
9  }
10 int main(){
11         int n,k,d,a,b,t=0;
```

```
12          scanf("%d%d",&n,&k);
13          for(int i=1;i<=n;i++)p[i]=i;
14          while(k--){
15                  scanf("%d%d%d",&d,&a,&b);
16                  if(a>n||b>n){
17                          t++;
18                          continue;
19                  }
20                  int x=find(a),y=find(b);
21                  if(x==y)t+=((r[b]+d+2)%3!=r[a]);
22                  else {
23                          p[x]=y;
24                          r[x]=(r[b]-r[a]+2+d)%3;
25                  }
26          }
27          printf("%d\n",t);
28          return 0;
29  }
```

1.10.3 简单区间合并

```
 1  int p[10000001];
 2  int find(int x){
 3      if(x!=p[x])x=find(p[x]);
 4      return p[x];
 5  }
 6  LL mpow(LL x, LL k){
 7      LL ret=1;
 8      while(k){
 9          if(k&1)ret=(ret*x)%1000000007;
10          k>>=1;
11          x=(x*x)%1000000007;
12      }
13      return ret;
14  }
15  int main(){
16      int n,m,x,y,l,r;
17      while(~scanf("%d%d",&n,&m)){
18          for(int i=n+1;i>0;i--)p[i]=i;
19          int ans=n;
20          while(m--){
21              scanf("%d%d",&l,&r);
22              x=find(l);
23              y=find(r+1);
24              if(x!=y){
25                  ans--;
26                  p[x]=y;
27              }
28          }
29          printf("%lld\n",mpow(26,ans));
30      }
31      return 0;
32  }
```

1.10.4 区间染色

求不可见颜色数

```
 1  #define N 10000
 2  struct rectan{
 3      int l,b,r,t;
 4      void get(){scanf("%d%d%d%d",&l,&b,&r,&t);}
 5  }rec[N];
 6  int p[N],cor[N],ans[N];
 7  void init(int x){
 8      for(int i=1;i<=x;i++){
 9          p[i]=i+1;
10          cor[i]=0;
11      }
```

```
}
int color(int l,int r,int c){
    if(!cor[l])cor[l]=c;
    if(p[l]<=r){
        p[l]=color(p[l],r,c);
    }
    return p[l];
}
int main(){
    int r,c,m,x,a,y,b;
    while(~scanf("%d%d%d",&c,&r,&m)){
        for(int i=1;i<=m;i++){
            ans[i]=0;
            rec[i].get();
        }
        for(int i=1;i<=r;i++){
            init(c);
            for(int j=m;j>0;j--)
                if(rec[j].b<=i&&rec[j].t>=i)
                    color(rec[j].l,rec[j].r,j);
            for(int j=c;j>0;j--)ans[cor[j]]=1;
        }
        int an=0;
        for(int i=m;i>0;i--)if(!ans[i])an++;
        printf("%d\n",an);
    }
}
```

1.11 小

1.11.1 最小表示法

可判断字符串同构，寻找字典序最小的循环串

```
int minishow(char *s){//返回最小表示的起始位置
        int i=0,j=1,k=0,n=strlen(s),m=(n>>1);
        while(i<m&&j<m&&k<m){
                if(s[i+k]==s[j+k])k++;
                else{
                        if(s[i+k]>s[j+k])i+=(k+1);//若是，则取的是最大表示<
                        else j+=(k+1);
                        if(i==j)j++;
                        k=0;
                }
        }
        return min(i,j);
}
```

1.11.2 构建指定序列的二叉树

```
#include <cstdio>
#include <stack>
#define MAXN 100010
using namespace std;

int h[MAXN];
int l[MAXN];
int r[MAXN];
int ls[MAXN];//左儿子
int rs[MAXN];//右儿子
int n;

bool check(int a, int b) {
        if(b<1||n<b) return false;
        return h[a]<h[b];
```

```
}

void erase(int x) {
        l[r[x]] = l[x];
        r[l[x]] = r[x];
}

int main() {
        scanf("%d", &n);
        stack<int> s;
        for(int x, i=1; i<=n; ++i) {
                scanf("%d", &x);
                h[x] = i;
                s.push(x);
        }
        for(int i=1; i<=n; ++i) {
                l[i] = i-1;
                r[i] = i+1;
        }
        int x;
        while(!s.empty()) {
                x = s.top();
                s.pop();
                if(check(x, l[x])) {
                        ls[x] = l[x];
                        erase(ls[x]);
                } else ls[x] = 0;
                if(check(x, r[x])) {
                        rs[x] = r[x];
                        erase(rs[x]);
                } else rs[x] = 0;
        }
        bool flag = false;
        s.push(x);
        while(!s.empty()) {
                x = s.top();
                s.pop();
                if(x==0) continue;
                if(flag) printf("␣");
                flag = true;
                printf("%d", x);
                s.push(rs[x]);
                s.push(ls[x]);
        }
        printf("\n");
}
```

1.11.3 heap 最小堆

```
#include <functional>
#include <queue>
#include <vector>
priority_queue< int, vector<int>, greater<int> > pri_qa;//最小堆
priority_queue< int, vector<int>, less<int> >pri_qb;//最大堆
#define cp(a,b) ((a)<(b))
struct heap {
    int h[N];
    int n,p,c;
    void ins(int e){
        for(p=++n;p>1&&cp(e,h[p>>1]);h[p]=h[p>>1],p>>=1);
        h[p]=e;
    }
    int del(int &e){
        if(!n)return 0;
        for(e=h[p=1],c=2;c<n&&cp(h[c+=(c<n-1&&cp(h[c+1],h[c]))],h[n]);h[p]=h[
           c],p = c,c<<=1);
        h[p]=h[n--];
        return 1;
```

```
19      }
20  };
21  heap p;
22  p.n=0;
```

1.11.4 多维曼哈顿距离

```
 1  #include <stdio.h>
 2  int n;
 3  double a[5],ma[32],mi[32];
 4  int main(){
 5      scanf("%d",&n);
 6      for(int i=0;i<32;i++)mi[i]=2000000000,ma[i]=-2000000000;
 7      while(n--){
 8          for(int i=0;i<5;i++)scanf("%lf",&a[i]);
 9          for(int i=0;i<32;i++){
10              double sum=0;
11              for(int j=0;j<5;j++){
12                  if((i>>j)&1)sum+=a[j];
13                  else sum-=a[j];
14              }
15              if(sum>ma[i])ma[i]=sum;
16              if(sum<mi[i])mi[i]=sum;
17          }
18      }
19      double ans=0;
20      for(int i=0;i<32;i++)
21          if(ans<ma[i]-mi[i])ans=ma[i]-mi[i];
22      printf("%.2lf\n",ans);
23      return 0;
24  }
```

1.11.5 前中推后序遍历

```
 1  #include <stdio.h>
 2  #include <string.h>
 3  char pre[27],in[27];
 4  void post(int s,int e ,int r){
 5          if(s>e)return;
 6          int i;
 7          for(i=s;i<=e&&pre[r]!=in[i];i++);
 8          post(s,i-1,r+1);
 9          post(i+1,e,r-s+i+1);
10          printf("%c",pre[r]);
11  }
12  int main(){
13          while(scanf("%s%s",pre,in)!=EOF){
14                  post(0,strlen(pre)-1,0);
15                  printf("\n");
16          }
17          return 0;
18  }
```

1.11.6 匹配次数

```
 1  void get_next(char t[],int len){
 2      for(int i=0,j=-1;i<=len;i++,j++){
 3          next[i]=j;
 4          while(j!=-1&&t[i]!=t[j])j=next[j];
 5      }
 6  }
 7  int kmp(char s[],char t[]){
 8      int len=strlen(t)-1,tim=0;
 9      for(int i=0,j=0;s[i];i++,j++){
10          if(j==len&&s[i]==t[j]){j=next[j];tim++;}
11          while(j!=-1&&s[i]!=t[j])j=next[j];
12      }
13      return tim;
```

```
14 }
15 //最大循环
16 //int ans=1;
17 //if(len%(len-next[len])==0)ans=len/(len-next[len]);
```

1.11.7 二维 RMQ

```
 1 for(int i=0;R[i]=1<<i,R[i]<MAXN;i++);
 2 void RM(){
 3     for(int i=0;i<n;i++)
 4         for(int j=0;j<m;j++)
 5             dp[0][0][i][j]=a[i][j];
 6     for(int i=0;R[i]<=n;i++)
 7         for(int j=0;R[j]<=m;j++)
 8             if(i||j)
 9                 for(int ii=0;ii+R[i]<=n;ii++)
10                     for(int jj=0;jj+R[j]<=m;jj++)
11                         if(i)dp[i][j][ii][jj]=min(dp[i-1][j][ii][jj],dp[i-1][
                               j][ii+R[i-1]][jj]);
12                         else dp[i][j][ii][jj]=min(dp[i][j-1][ii][jj],dp[i][j
                               -1][ii][jj+R[j-1]]);
13 }
14 int Q(int r1,int c1,int r2,int c2){
15     int i=0,j=0,h=(r2-r1+1)>>1,w=(c2-c1+1)>>1;
16     while(R[i]<=h)i++;
17     while(R[j]<=w)j++;
18     int ii=r2-R[i]+1,jj=c2-R[j]+1;
19     return min(min(dp[i][j][ii][c1],dp[i][j][r1][jj]),min(dp[i][j][r1][c1],dp
           [i][j][ii][jj]));
20 }
```

Chapter 2

Computation Geometry

2.1 矩形切割

2.1.1 二分计数

```
#include <cstdio>
#include <iostream>
using namespace std;
typedef long long LL;
#define D 2
struct cube{
    int s[D],e[D];
    int c;
    cube(){}
    cube(cube &t,int m,int c):c(c){
        for(int i=0;i<D;i++){
            s[i]=t.s[i]-m;
            e[i]=t.e[i]+m;
        }
    }
    LL area(cube &t){
        if(!cross(t))return 0;
        LL k=1;
        for(int i=0;i<D;i++)
            k*=(LL)(min(e[i],t.e[i])-max(s[i],t.s[i]));
        return k;
    }
    bool cross(cube &t){
        for(int i=0;i<D;i++)
            if(s[i]>=t.e[i]||e[i]<=t.s[i])return 0;
        return 1;
    }
    bool contains(cube &t){
        for(int i=0;i<D;i++)
            if(s[i]>t.s[i]||e[i]<t.e[i])return 0;
        return 1;
    }
}p[30],big;
cube c[1000],d[1000];
int q;
struct CUT{
    int cn,dn;
    void broke(cube &a,cube &b){
        cube t;
        for(int i=0;i<D;i++){
            if(b.s[i]>a.s[i]&&b.s[i]<a.e[i]){
                t=a;
                t.e[i]=b.s[i];
                d[dn++]=t;
                a.s[i]=b.s[i];
            }
            if(b.e[i]>a.s[i]&&b.e[i]<a.e[i]){
                t=a;
                t.s[i]=b.e[i];
                d[dn++]=t;
                a.e[i]=b.e[i];
            }
        }
    }
    void insert(cube &t){
        dn=0;
        for(int i=0;i<cn;i++){
            if(t.cross(c[i]))broke(c[i],t);
            else d[dn++]=c[i];
        }
        cn=0;
        c[cn++]=t;
        for(int i=0;i<dn;i++)c[cn++]=d[i];
```

```
64      }
65      void push(int m,int c){
66          for(int i=0;i<q;i++){
67              cube t(p[i],m,c);
68              insert(t);
69          }
70      }
71      LL area(cube &t){
72          LL ans=0;
73          for(int i=0;i<cn;i++)
74             ans+=c[i].area(t);
75          return ans;
76      }
77      void refush(){
78          int n=0;
79          for(int i=0;i<cn;i++)
80              if(c[i].c)c[n++]=c[i];
81          cn=n;
82      }
83  }cut;
84  void solve(){
85      LL n;
86      cin>>n;
87      int l=0,r=max(big.e[0],big.e[1]);
88      while(l<r){
89          cut.cn=0;
90          int m=(l+r)>>1;
91          cut.push(m,1);
92          if(cut.area(big)-q<n)l=m+1;
93          else r=m;
94      }
95      cut.cn=0;
96      cut.push(l-1,1);
97      n-=(cut.area(big)-q);
98      cut.cn=0;
99      cut.push(l,1);
100     cut.push(l-1,0);
101     cut.refush();
102     cube res=big;
103     l=0,r=big.e[1];
104     while(l<r){
105         res.e[1]=(l+r)>>1;
106         if(cut.area(res)<n)l=res.e[1]+1;
107         else r=res.e[1];
108     }
109     res.e[1]=l-1;
110     n-=cut.area(res);
111     res.s[1]=l-1;
112     res.e[1]=l;
113     l=0,r=big.e[0];
114     while(l<r){
115         res.e[0]=(l+r)>>1;
116         if(cut.area(res)<n)l=res.e[0]+1;
117         else r=res.e[0];
118     }
119     printf("%d␣%d\n",res.e[1],l);
120 }
121 int main(){
122     while(scanf("%d%d%d",&big.e[1],&big.e[0],&q),big.e[0]||big.e[1]||q){
123         big.s[0]=big.s[1]=0;
124         for(int i=0;i<q;i++){
125             for(int j=D-1;j>=0;j--){
126                 scanf("%d",&p[i].e[j]);
127                 p[i].s[j]=p[i].e[j]-1;
128             }
129         }
130         int Q;
```

```
131         scanf("%d",&Q);
132         while(Q--)solve();
133         puts("-");
134     }
135     return 0;
136 }
```

2.2 圆弧的离散化

2.2.1 圆面积并

```
 1 ct dd eps=1e-8;
 2 ct dd pi=acos(-1.0);
 3 struct pt{
 4     dd x,y;
 5     pt(){}
 6     pt(dd x,dd y):x(x),y(y){}
 7     void get(){scanf("%lf%lf",&x,&y);}
 8     pt op+(cpt p)ct{rt pt(x+p.x,y+p.y);}
 9     pt op-(cpt p)ct{rt pt(x-p.x,y-p.y);}
10     dd op^(cpt p)ct{rt x*p.y-y*p.x;}
11     dd abs()ct{rt sqrt(x*x+y*y);}
12 };
13 dd eve[3000];
14 int e,id[3000];
15 struct cc{
16     pt o;
17     dd r;
18     void get(){
19         o.get();
20         scanf("%lf",&r);
21     }
22     bool in(ct cc &c){
23         rt (o-c.o).abs()<r-c.r+eps;
24     }
25     pt get_pt(dd v)ct{
26         rt o+pt(r*cos(v),r*sin(v));
27     }
28     void c_c(ct cc& c){
29         pt dir=c.o-o;
30         dd d=dir.abs();
31         if(r+c.r<d+eps)rt;
32         dd v=atan2(dir.y,dir.x);//(-pi,pi]
33         dd add=acos(((d*d+r*r-c.r*c.r)/d)/(r+r));//[0,pi]
34         eve[e++]=v-add,eve[e++]=v+add;
35     }
36 }c[1010];
37 bool cmp(int a,int b){rt eve[a]<eve[b];}
38 struct min_cc{
39     dd area;
40     cc *c;
41     int n;
42     min_cc(){}
43     min_cc(cc _c[],int _n){
44         area=0;
45         c=_c,n=_n;
46         for(int i=0;i<n;i++)
47         for(int j=0;j<n;j++)
48             if(i>=0&&i!=j&&c[j].in(c[i]))
49                 c[i--]=c[--n];
50         for(int i=0;i<n;i++){
51             e=0;
52             for(int j=0;j<n;j++)
53                 if(i!=j)c[i].c_c(c[j]);
54             radar(c[i]);
55         }
56         area*=.5;
```

```
    }
    void radar(ct cc &c){
        for(int i=0,t=e;i<t;i+=2){
            if(eve[i]<-pi){
                eve[i]+=pi*2;
                eve[i+1]+=pi*2;
            }
            if(eve[i+1]>pi){
                eve[e++]=-pi;
                eve[e++]=eve[i+1]-pi*2;
                eve[i+1]=pi;
            }
        }
        eve[e++]=-pi,eve[e++]=pi;
        for(int i=0;i<e;i++)id[i]=i;
        sort(id,id+e,cmp);
        int cnt=0;
        for(int i=0;i<e;i++){
            if(cnt==1){
                int a=id[i-1],b=id[i];
                dd v=eve[b]-eve[a];
                if(v>eps)
                    area+=(c.get_pt(eve[a])^c.get_pt(eve[b]))+c.r*c.r*(v-sin(
                        v));
            }
            if(id[i]&1)cnt--;
            else cnt++;
        }
    }
};
int main(){
    int n;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        c[i].get();
        if(c[i].r<eps)i--,n--;
    }
    printf("%.3lf\n",min_cc(c,n).area);
    rt 0;
}
```

2.2.2 圆面积交

同上，输出覆盖了 [1，n] 次的面积, 各种题目变形的时候注意数据预处理。

```
struct cc{
    pt o;
    dd r;
    void get(){
        o.get();
        scanf("%lf",&r);
    }
    pt get_pt(dd v)ct{
        rt o+pt(r*cos(v),r*sin(v));
    }
    void c_c(ct cc& c){
        pt dir=c.o-o;
        dd d=dir.abs();
        if(c.r-r>d-eps){
            eve[e++]=-pi,eve[e++]=pi;rt;
        }
        if(r-c.r>d-eps||r+c.r<d+eps)rt;
        dd v=atan2(dir.y,dir.x);//(-pi,pi]
        dd add=acos(((d*d+r*r-c.r*c.r)/d)/(r+r));//[0,pi]
        eve[e++]=v-add,eve[e++]=v+add;
    }
}c[1010];
dd s2[1010];
void radar(ct cc &c){
```

```
25      for(int i=0,t=e;i<t;i+=2){
26          if(eve[i]<-pi){
27              eve[i]+=pi*2;
28              eve[i+1]+=pi*2;
29          }
30          if(eve[i+1]>pi){
31              eve[e++]=-pi;
32              eve[e++]=eve[i+1]-pi*2;
33              eve[i+1]=pi;
34          }
35      }
36      eve[e++]=-pi,eve[e++]=pi;
37      for(int i=0;i<e;i++)id[i]=i;
38      sort(id,id+e,cmp);
39      int cnt=0;
40      for(int i=0;i<e;i++){
41          if(cnt){
42              int a=id[i-1],b=id[i];
43              dd v=eve[b]-eve[a];
44              if(v>eps)
45                  s2[cnt]+=(c.get_pt(eve[a])^c.get_pt(eve[b]))+c.r*c.r*(v-sin(v
                        ));
46          }
47          if(id[i]&1)cnt--;
48          else cnt++;
49      }
50  }
51  int main(){
52      int n;
53      scanf("%d",&n);
54      for(int i=0;i<n;i++)c[i].get();
55      memset(s2,0,sizeof(s2));
56      for(int i=0;i<n;i++){
57          e=0;
58          for(int j=0;j<n;j++)
59              if(i!=j)
60                  c[i].c_c(c[j]);
61          radar(c[i]);
62      }
63      for(int i=1;i<=n;i++)
64          printf("[%d] = %.3lf\n",i,(s2[i]-s2[i+1])*.5);
65      rt 0;
66  }
```

2.2.3 矩形框类有若干相离的圆，求还能放入的最大圆

二分半径，膨胀圆，通过圆弧离散化可以判断是否有空间。

```
1   dd eve[200];
2   int e,id[200];
3   struct cc{
4       pt o;
5       dd r;
6       void get(){
7           o.get();
8           scanf("%lf",&r);
9       }
10      void c_c(ct cc& c){
11          pt dir=c.o-o;
12          dd d=dir.abs();
13          if(r+c.r<d+eps)rt;//相离，不处理全包含
14          if(c.r>r-eps&&c.r-r>d+eps){//包含，处理全包含
15              eve[e++]=-pi,eve[e++]=pi;
16              rt;
17          }
18          dd v=atan2(dir.y,dir.x);//(-pi,pi]
19          dd add=acos(((d*d+r*r-c.r*c.r)/d)/(r+r));//[0,pi]
```

```
20          eve[e++]=v-add,eve[e++]=v+add;//(-2pi,2pi]&&a<b&&(b-a)[0,pi]
21      }
22      void c_c(cpt a,cpt b){//覆盖顺时针方向的弧a->b
23          dd d=(o.X(a,b)/(b-a).abs());
24  //        if(d>r-eps||d<-r+eps)rt不处理全包含;//
25          if(d>r-eps)rt;//离
26          if(d<-r+eps){//反向离，处理全包含
27              eve[e++]=-pi,eve[e++]=pi;
28              rt;
29          }
30          pt dir=(a-b).TR();
31          dd v=atan2(dir.y,dir.x),add=acos(d/r);
32          eve[e++]=v-add,eve[e++]=v+add;
33      }
34  }c[100],cp[100];
35  dd W,H,R;
36  bool cmp(int a,int b){
37      rt eve[a]<eve[b];
38  }
39  struct min_cc{
40      cc *c;
41      int n;
42      bool res;
43      min_cc(){}
44      min_cc(cc _c[],int _n){
45          res=0;
46          c=_c,n=_n;
47          for(int i=0;i<n;i++){
48              e=0;
49              for(int j=0;j<n;j++)
50                  if(i!=j)c[i].c_c(c[j]);
51              pt p[5];
52              p[4]=p[0]=pt(R,R),p[1]=pt(W-R,R),p[2]=pt(W-R,H-R),p[3]=pt(R,H-R);
53              for(int j=0;j<4;j++)c[i].c_c(p[j],p[j+1]);
54              res|=radar(c[i]);
55              if(res==1)rt;
56          }
57      }
58      bool radar(ct cc &c){
59          for(int i=0,t=e;i<t;i+=2){
60              if(eve[i]<-pi){
61                  eve[i]+=pi*2;
62                  eve[i+1]+=pi*2;
63              }
64              if(eve[i+1]>pi){
65                  eve[e++]=-pi;
66                  eve[e++]=eve[i+1]-pi*2;
67                  eve[i+1]=pi;
68              }
69          }
70          eve[e++]=-pi,eve[e++]=pi;
71          for(int i=0;i<e;i++)id[i]=i;
72          sort(id,id+e,cmp);
73          int cnt=0;
74          for(int i=0;i<e;i++){
75              if(cnt==1&&eve[id[i]]-eve[id[i-1]]>eps)rt 1;
76              if(id[i]&1)cnt--;
77              else cnt++;
78          }
79          rt 0;
80      }
81  };
82  int n;
83  bool judge(dd r){
84      R=r;
85      for(int i=0;i<n;i++){
86          cp[i]=c[i];
```

```
87          cp[i].r+=r;
88      }
89      rt n==0||min_cc(cp,n).res;
90  }
91  int main(){
92      int T;
93      scanf("%d",&T);
94      while(T--){
95          scanf("%lf%lf",&W,&H);
96          scanf("%d",&n);
97          for(int i=0;i<n;i++)c[i].get();
98          dd l=0,r=(min(H,W))*.5;
99          while(l+(1e-6)<r){
100             dd m=(l+r)*.5;
101             if(judge(m))l=m;
102             else r=m;
103         }
104         printf("%.10lf\n",r);
105     }
106     return 0;
107 }
```

2.3 随机增量

2.3.1 点集最小周长三角形

grid 与 key 作为成员函数，并且表示点与格点两种信息，使用时有 key()grid() 的错误, 重复 hash。
i 点如果可以更新最优值，更新 hash 表，否则加入 i 点。

```
1  #define F 971
2  #define H 65535
3  #define ct const
4  #define cpt ct pt&
5  #define rt return
6  #define dd double
7  #define op operator
8  dd ans;
9  struct pt{
10     int x,y;
11     void get(){scanf("%d%d",&x,&y);}
12     void out(){printf("%d␣%d\n",x,y);}
13     pt(int x=0,int y=0):x(x),y(y){}
14     pt op+(cpt p)ct{rt pt(x+p.x,y+p.y);}
15     pt op-(cpt p)ct{rt pt(x-p.x,y-p.y);}
16     dd abs(){rt sqrt(1.0*x*x+1.0*y*y);}
17     pt grid()ct{
18         int size=ans/2;
19         rt pt(x/size+1,y/size+1);
20     }
21     int key(){
22         rt (x*F+y)&H;
23     }
24 }p[20010],dir[9];
25 int n;
26 int head[H];
27 pt ss[20010];
28 int ID;
29 void insert(int id){
30     int key=p[id].grid().key();
31     ss[ID].x=id;
32     ss[ID].y=head[key];
33     head[key]=ID++;
34 }
35 void refresh(int n){
36     memset(head,-1,sizeof(head));
37     ID=0;
```

```
    for(int i=0;i<n;i++){
        insert(i);
    }
}

int eve[20010];
int e;
void get_pt(cpt t){
    pt grid=t.grid();
    e=0;
    for(int i=0;i<9;i++){
        int key=(grid+dir[i]).key();
        for(int j=head[key];~j;j=ss[j].y){
            if((t-p[ss[j].x]).abs()<ans/2){
                eve[e++]=ss[j].x;
            }
        }
    }
}

dd calc(cpt a,cpt b,cpt c){
    rt (a-b).abs()+(b-c).abs()+(a-c).abs();
}

int main(){
    srand(time(0));
    for(int i=0;i<3;i++)
        for(int j=0;j<3;j++){
            dir[i*3+j]=pt(i-1,j-1);
        }
    int T;
    scanf("%d",&T);
    while(T--){
        scanf("%d",&n);
        for(int i=0;i<n;i++){
            p[i].get();
        }
        for(int i=0;i<n;i++){
            swap(p[i],p[rand()%n]);
        }
        ans=calc(p[0],p[1],p[2]);
        refresh(3);
        for(int i=3;i<n;i++){
            get_pt(p[i]);
            dd now=1e100;
            for(int j=0;j<e;j++){
                if(i==eve[j])continue;
                for(int k=j+1;k<e;k++){
                    if(i==eve[k]||eve[j]==eve[k])continue;
                    now=min(now,calc(p[i],p[eve[j]],p[eve[k]]));
                }
            }
            if(now<ans){
                ans=now;
                refresh(i+1);
            }else{
                insert(i);
            }
        }
        printf("%.3lf\n",ans);
    }
}
```

2.4 扫描线

2.4.1 set，光源能照亮周围的木棒的数量

以光线和木棒的交点到光源的距离作事件比较，极扫描

```
 1 struct pt{
 2     dd x,y;
 3     void get(){scanf("%lf%lf",&x,&y);}
 4     pt(dd x=0,dd y=0):x(x),y(y){}
 5     pt op+(cpt p)ct{rt pt(x+p.x,y+p.y);}
 6     pt op-(cpt p)ct{rt pt(x-p.x,y-p.y);}
 7     pt op*(dd v)ct{rt pt(x*v,y*v);}
 8     dd op^(cpt p)ct{rt x*p.y-y*p.x;}
 9     pt(cpt a,cpt b,cpt c,cpt d){
10         dd t=(d-c^a-c)/(b-a^d-c);
11         *this=a*(1-t)+b*t;
12     }
13     double abs2(){rt x*x+y*y;}
14 }o,A,B,p[20010];
15
16 struct event{
17     pt *a,*b;
18     dd v;
19     int id;
20     bool type;
21     event(){}
22     event(pt *a,pt *b,int id,dd v,bool type):a(a),b(b),id(id),v(v),type(type)
          {}
23     bool op<(ct event &e)ct{
24         if(a==e.a&&b==e.b)rt 0;
25         rt pt(A,B,*a,*b).abs2()<pt(A,B,*e.a,*e.b).abs2();
26     }
27     bool judge(){
28         if(type==0)rt 0;
29         pt t(pt(),pt(1,0),*a,*b);
30         rt t.x<0&&t.y<a->y&&t.y>b->y;
31     }
32 }eve[20010];
33 set<event> radar;
34 bool v[10010];
35 bool cmp(ct event &a,ct event &b){
36     rt a.v<b.v;
37 }
38 int main(){
39     int n;
40     while(~scanf("%d",&n)){
41         o.get();
42         int e=0;
43         for(int i=0;i<n;i++){
44             p[i].get();
45             p[i]=p[i]-o;
46             p[i+n].get();
47             p[i+n]=p[i+n]-o;
48             if((p[i]^p[i+n])<0)swap(p[i],p[i+n]);
49             eve[e++]=event(&p[i],&p[i+n],i,atan2(p[i].y,p[i].x),1);
50             eve[e++]=event(&p[i],&p[i+n],i,atan2(p[i+n].y,p[i+n].x),0);
51         }
52         sort(eve,eve+e,cmp);
53         fill(v,v+n,0);
54         radar.clear();
55         B=pt(-1,0);
56         for(int i=0;i<e;i++)去周期{\\
57             if(eve[i].judge()){
58                 radar.insert(eve[i]);
59             }
60         }
```

```
61          for(int i=0;i<e;i++){
62              if(eve[i].type){
63                  B=*(eve[i].a);
64                  radar.insert(eve[i]);
65              }else{
66                  B=*(eve[i].b);
67                  radar.erase(eve[i]);
68              }
69              if(!radar.empty()){
70                  v[radar.begin()->id]=1;
71              }
72          }
73          int cnt=0;
74          for(int i=0;i<n;i++)if(v[i])cnt++;
75          printf("%d\n",cnt);
76      }
77      rt 0;
78  }
```

2.4.2 set，最近圆对

最近圆对，水平扫描，垂直比较，直接判两个圆是否相交

```
1   typedef long long LL;
2   const double eps=1e-8;
3   struct cc{
4       double x,y,r;
5       void get(){scanf("%lf%lf%lf",&x,&y,&r);}
6       bool operator<(const cc &p)const{return y<p.y||y==p.y&&x<p.x;}
7   }c[50010];
8   struct event{
9       double x;
10      int id;
11      bool v;
12      event(){}
13      event(double x,int id,bool v):x(x),id(id),v(v){}
14      bool operator <(const event &p)const{return x<p.x;}
15  }eve[100010];
16  bool inter(cc a,cc b,double &r){
17      return (a.x-b.x)*(a.x-b.x)+(a.y-b.y)*(a.y-b.y)<(a.r+b.r+r+r)*(a.r+b.r+r+r
            )+eps;
18  }
19  set<cc>line;
20  typedef set<cc>::iterator IT;
21  int T,n,e;
22  bool judge(double d){
23      line.clear();
24      e=0;
25      for(int i=0;i<n;i++){
26          eve[e++]=event(c[i].x-d-c[i].r,i,1);
27          eve[e++]=event(c[i].x+d+c[i].r,i,0);
28      }
29      sort(eve,eve+e);
30      for(int i=0;i<e;i++){
31          if(eve[i].v){
32              IT it=line.insert(c[eve[i].id]).first,s=it,e=it;
33              if(s--!=line.begin())
34                  if(inter(*s,*it,d))return 1;
35              if(++e!=line.end())
36                  if(inter(*e,*it,d))return 1;
37          }else line.erase(c[eve[i].id]);//此处有，要判断bugerase
38      }
39      return 0;
40  }
41
42  int main(){
43      scanf("%d",&T);
```

```
    while(T--){
        scanf("%d",&n);
        for(int i=0;i<n;i++)c[i].get();
        double l=0,r=100000;
        while(l+eps<r){
            double m=(l+r)/2;
            if(judge(m))r=m;
            else l=m;
        }
        printf("%lf\n",l+r);
    }
}
```

2.4.3 set，极扫描圆

求点集的联通分量，之间有圆为障碍物，枚举点为中心极扫描，比较为切点到圆的

```
struct point{
    double x,y;
    int id;
    void get(int t){scanf("%lf%lf",&x,&y);id=t;}
}p[1001];
struct circle{
    point p;
    double r;
    void get(){scanf("%lf%lf%lf",&p.x,&p.y,&r);}
}c[1001];
int f[1001],n,m;
struct event{
    double v,d;
    int id;
    event(){}
    event(double v,double d,int id):v(v),d(d),id(id){}
    bool operator<(const event &e)const{return v<e.v;}
}eve[6003];
inline double dis2(point &a,point &b){
    return (a.x-b.x)*(a.x-b.x)+(a.y-b.y)*(a.y-b.y);
}
int find(int x){
    if(x!=f[x])f[x]=find(f[x]);
    return f[x];
}
set<double> line;
int main(){
    while(~scanf("%d%d",&n,&m)){
        for(int i=0;i<n;i++)p[i].get(i),f[i]=i;
        for(int i=0;i<m;i++)c[i].get();
        for(int i=0;i<n;i++){
            swap(p[0],p[i]);
            int e=0;
            for(int j=1;j<n;j++)
                if(find(i)!=find(j))eve[e++]=event(atan2(p[j].y-p[0].y,p[j].x
                  -p[0].x),sqrt(dis2(p[0],p[j])),p[j].id);
            for(int j=0;j<m;j++){
                double  dv=asin(c[j].r/sqrt(dis2(p[0],c[j].p))),
                        ds=sqrt(dis2(p[0],c[j].p)-c[j].r*c[j].r),//切线长度
                        tv=atan2(c[j].p.y-p[0].y,c[j].p.x-p[0].x);
                eve[e++]=event(tv-dv,ds,-2);
                eve[e++]=event(tv+dv,ds,-1);
                if(tv-dv<-pi){
                    eve[e++]=event(tv-dv+2*pi,ds,-2);
                    eve[e++]=event(tv+dv+2*pi,ds,-1);
                }
                if(tv+dv>pi){
                    eve[e++]=event(tv-dv-2*pi,ds,-2);
                    eve[e++]=event(tv+dv-2*pi,ds,-1);
```

```
                }
            }
            sort(eve,eve+e);
            line.clear();
            for(int j=0;j<e;j++){
                if(eve[j].id==-2)line.insert(eve[j].d);
                else if(eve[j].id==-1)line.erase(eve[j].d);
                else {
                    int x=find(i),y=find(eve[j].id);
                    if(x!=y&&line.size()==0||eve[j].d<*line.begin())f[x]=find
                       (y);
                }
            }
        }
        int cnt=-1;
        for(int i=0;i<n;i++)if(i==find(i))cnt++;
        printf("%d\n",cnt);
    }
    return 0;
}
```

2.4.4 set，求有多少个不被包含的圆

注意 sqrt 里面为负数

```
double L;
const double eps=1e-8;
struct circle{
    double x,y,r;
    bool v;
    void get(){scanf("%lf%lf%lf",&r,&x,&y);}
    double Y(bool f){
        double t=sqrt(r*r-(L-x)*(L-x)+eps);//
        return y+(f?t:-t);
    }
}c[40001];
struct node{
    int p;
    bool v;
    node(){}
    node(int p,bool v):p(p),v(v){}
    bool operator<(const node &other)const{
        double y1=c[p].Y(v)*100,y2=c[other.p].Y(other.v)*100;
        return y1>y2||y1==y2&&v>other.v;
    }
};
set<node> line;
typedef set<node>::iterator IT;
struct event{
    double x,y;
    int p;
    bool v;
    event(){}
    event(double x,double y,int p,bool v):x(x),y(y),p(p),v(v){}
    bool operator<(const event &other)const{
        return x<other.x||x==other.x&&y>other.y;
    }
}eve[80002];
int main(){
    int m=0,n,cnt=0;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        c[i].get();
        eve[m++]=event(c[i].x-c[i].r,c[i].y,i,1);
        eve[m++]=event(c[i].x+c[i].r,c[i].y,i,0);
    }
    sort(eve,eve+m);
```

```
43      for(int i=0;i<m;i++){
44          L=eve[i].x;
45          if(eve[i].v){
46              IT it=line.insert(node(eve[i].p,0)).first,st=it,ed=it;
47              if(st--==line.begin()||++ed==line.end())c[it->p].v=1,cnt++;
48              else{
49                  if(st->p==ed->p)c[it->p].v=0;
50                  else c[it->p].v=c[st->p].v&&c[ed->p].v,cnt+=c[it->p].v;
51              }
52              line.insert(node(eve[i].p,1));
53          }else{
54              line.erase(node(eve[i].p,0));
55              line.erase(node(eve[i].p,1));
56          }
57      }
58      printf("%d\n",cnt);
59      for(int i=0;i<n;i++)if(c[i].v)printf("%d␣",i+1);
60      return 0;
61  }
```

2.4.5 BIT，求各个矩形中的点的数目

水平扫描线，求矩形中的点的数目，结构为树状数组

```
 1  struct pt{
 2      int x,y,z;
 3      void get(){
 4          double t;
 5          scanf("%d%d%lf",&x,&y,&t);
 6          t*=100.0;
 7          z=(int)(t>0?t+0.1:t-0.1);
 8      }
 9      bool operator<(const pt &p)const{return y<p.y;}
10  }p[60010];
11  struct cube{
12      int x1,y1,z1,x2,y2,z2;
13      void get(){
14          scanf("%d%d%d%d",&x1,&y1,&x2,&y2);
15          y1--;
16          z1=z2=0;
17      }
18      void out(){
19          printf("%.2lf/%d\n",(double)z1/100.0,z2);
20      }
21  }q[20010];
22
23  #define get_id(a) ((a)=(lower_bound(x,x+nx,a)-x+1))
24  int x[100010],y[40010],id[40010],v[100010],c[100010],nx,ny;
25  bool cmp(int a,int b){
26      return y[a]<y[b];
27  }
28  void add(int i,int x){
29      while(i<=nx){
30          v[i]+=x;
31          c[i]++;
32          i+=i&-i;
33      }
34  }
35  void calc(int i,int &x,int &y){
36      x=y=0;
37      while(i){
38          x+=v[i];
39          y+=c[i];
40          i-=i&-i;
41      }
42  }
43  int main(){
```

```
    int n,m;
    while(~scanf("%d%d",&n,&m)){
        nx=0,ny=0;
        for(int i=0;i<n;i++){
            p[i].get();
            x[nx++]=p[i].x;
        }
        for(int i=0;i<m;i++){
            q[i].get();
            x[nx++]=q[i].x1;
            x[nx++]=q[i].x2;
            id[ny]=ny;
            y[ny++]=q[i].y1;
            id[ny]=ny;
            y[ny++]=q[i].y2;
        }
        sort(x,x+nx);
        nx=unique(x,x+nx)-x;
        for(int i=0;i<n;i++)get_id(p[i].x);
        for(int i=0;i<m;i++){
            get_id(q[i].x1);
            get_id(q[i].x2);
        }
        fill(v,v+nx+1,0);
        fill(c,c+nx+1,0);
        sort(p,p+n);
        sort(id,id+ny,cmp);
        for(int i=0,j=0;i<ny;i++){
            int t=id[i];
            while(j<n&&p[j].y<=y[t]){
                add(p[j].x,p[j].z);
                j++;
            }
            int t1,t2,t3,t4;
            calc(q[t>>1].x1-1,t1,t2),calc(q[t>>1].x2,t3,t4);
            q[t>>1].z1+=(t&1)?t3-t1:t1-t3;
            q[t>>1].z2+=(t&1)?t4-t2:t2-t4;
        }
        for(int i=0;i<m;i++)q[i].out();
    }
    return 0;
}
```

2.5 模拟退火

2.5.1 最小球覆盖

```
//最小球覆盖，要往离自己最远的方向走。爬山法
const double eps=1e-10;
struct point{
    double x,y,z;
        ...
}p[110];
int main(){
    int n;
    while(~scanf("%d",&n)){
        for(int i=0;i<n;i++)p[i].get();
        point o=p[0];
        double ans=1e10,step=10000;
        while(step>eps){
            double max_d=0;
            int id;
            for(int i=0;i<n;i++){
                double dis=(p[i]-o).abs2();
                if(dis>max_d){
                    id=i;
```

```
                    max_d=dis;
                }
            }
            if(max_d<ans)ans=max_d;
            if(fabs(max_d)<eps)break;
            o=o+(p[id]-o)*(step/sqrt(max_d));
            step*=0.999;
        }
        o.out();
    }
    return 0;
}
```

2.5.2 区域内最优点，并行

注意淘汰区域外的尝试。可能忽略边界上的解，必要的话需要枚举边界点
(P,L,delta,eps)
区域内最近距离最远，点数 1000，题目精度要求 0.1，（10 20 0.9 0.01 ）
费马点，点数 100，要求保留整数，（1 3 0.7 0.1 ）
点到圆的视角相同，圆数 3，使用方差做评估函数，精度 0.01，（5 10 0.8 0.001 ）

```
const int P=5,L=10;
dd rand_f(){rt 1.0*rand()/(RAND_MAX-1);}//[0,1]
struct pt{
    dd x,y,r;
    pt(){}
    pt(dd x,dd y):x(x),y(y),r(0){}
    ...
    void get(){scanf("%lf%lf%lf",&x,&y,&r);}
    void out(){printf("%.2lf %.2lf\n",x,y);}
    bool pre(){rt x||y||r;}
}pos[P],c[3],s,e;
pt rand_pt(dd X,dd Y){
    rt pt(rand_f()*X,rand_f()*Y);
}
pt rand_dir(dd k){
    dd v=pi*2*rand_f();
    rt pt(k*cos(v),k*sin(v));
}
int X,Y,n;
dd calc(cpt t){
    dd a[3];
    dd sum=0,res=0;
    for(int i=0;i<3;i++){
        a[i]=(c[i]-t).abs()/c[i].r;
        sum+=a[i];
    }
    sum/=3.0;
    for(int i=0;i<3;i++)
        res+=(sum-a[i])*(sum-a[i]);
    rt res;
}
bool check(cpt t){
    rt t.x>=s.x&&t.x<=e.x&&t.y>=s.y&&t.y<=e.y;
}
int main(){
    srand(time(0));
    while(true){
        for(int i=0;i<3;i++){
            c[i].get();
            if(i){
                s.x=min(s.x,c[i].x);
                s.y=min(s.y,c[i].y);
                e.x=max(e.x,c[i].x);
                e.y=max(e.y,c[i].y);
            }else s=e=c[0];
        }
        if((c[0].pre()||c[1].pre()||c[2].pre())==0)break;
        e=e-s;
```

```
49          for(int i=0;i<P;i++){
50              pos[i]=s+rand_pt(e.x,e.y);
51              pos[i].r=calc(pos[i]);
52          }
53          dd step=max(e.x,e.y);
54          while(step>0.001){
55              for(int i=0;i<P;i++)
56                  for(int j=0;j<L;j++){
57                      pt t=pos[i]+rand_dir(step);
58                      if(!check(t))continue;
59                      t.r=calc(t);
60                      if(t.r<pos[i].r)pos[i]=t;
61                  }
62              step*=0.8;
63          }
64          int id=0;
65          for(int i=1;i<P;i++)
66              if(pos[i].r<pos[id].r)id=i;
67          if(pos[id].r<0.001)pos[id].out();
68          else puts("No solution");
69          }
70          return 0;
71      }
```

2.6 Simpson 公式

数据大时可以找事件点，数据小或事件点难以确定可以二分区间利用 Simpson 公式自适应找寻事件点。

2.6.1 拟柱体剖分求体积

x 轴扫描面，求水平棱镜和竖直棱镜的体积交，离散 x 相邻 x 形状为拟柱体，点数少，所以 O(n*n) 处理，这种图形在之间的要特别注意两边的边界情况。

```
1  #define Max(a,b) (a=((a)>(b)?(a):(b)))
2  #define Min(a,b) (a=((a)<(b)?(a):(b)))
3  const double eps=1e-8;
4  int sig(double x){
5      return (x>eps)-(x<-eps);
6  }
7  struct pt{
8      int x,y;
9      void get(){
10         scanf("%d%d",&x,&y);
11     }
12 }a[110],b[110];
13 int na,nb;
14 inline double length(pt p[],int n,double x){
15     double l=200,r=-200;
16     for(int i=0;i<n;i++){
17         pt a=p[i],b=p[i+1];
18         if(a.x==b.x)continue;
19         if(x+eps>a.x&&x<b.x+eps||x+eps>b.x&&x<a.x+eps){
20             double y=1.0*(a.y-b.y)/(a.x-b.x)*(x-a.x)+a.y;
21             Min(l,y);
22             Max(r,y);
23         }
24     }
25     if(l>r)return 0;
26     return r-l;
27 }
28 inline double area(double x){
29     return length(a,na,x)*length(b,nb,x);
30 }
31 int eve[200];
32 int main(){
```

```
33      while(scanf("%d%d",&na,&nb),na||nb){
34          int e=0;
35          int l1=200,r1=-200,l2=200,r2=-200;
36          for(int i=0;i<na;i++){
37              a[i].get();
38              Max(r1,a[i].x);
39              Min(l1,a[i].x);
40              eve[e++]=a[i].x;
41          }
42          for(int i=0;i<nb;i++){
43              b[i].get();
44              Max(r2,b[i].x);
45              Min(l2,b[i].x);
46              eve[e++]=b[i].x;
47          }
48          a[na]=a[0];
49          b[nb]=b[0];
50          int l=Max(l1,l2),r=Min(r1,r2);
51          for(int i=0;i<e;i++){
52              if(eve[i]<l||eve[i]>r)
53                  eve[i--]=eve[--e];
54          }
55          sort(eve,eve+e);
56          e=unique(eve,eve+e)-eve;
57          double v=0,s1=area(eve[0]);
58          for(int i=1;i<e;i++){
59              double s2=area(eve[i]);
60              v+=(s1+s2+4*area((eve[i]+eve[i-1])*.5))*(eve[i]-eve[i-1]);
61              s1=s2;
62          }
63          printf("%lf\n",v/6.0);
64      }
65      return 0;
66  }
```

2.6.2 梯形剖分求面积

也可以用拟柱体。

```
1   struct seg{
2       pt a,b;
3       dd k;
4       void get(){
5           a.get();
6           b.get();
7           k=(b.y-a.y)/(b.x-a.x);
8       }
9       dd high(dd x){
10          rt a.y+k*(x-a.x);
11      }
12      dd A2(){
13          rt (a.y+b.y)*(b.x-a.x);
14      }
15  }s[110];
16  int n,k;
17  dd length(dd x){
18      dd low=0;
19      for(int i=0;i<k;i++)
20          if(x>s[i].a.x-eps&&x<s[i].b.x+eps)
21              low=max(low,s[i].high(x));
22      rt max(0.0,s[k].high(x)-low);
23  }
24  dd area(dd l,dd r){
25      rt (length(l)+length(r))*(r-l);
26  }
27  dd dfs(dd l,dd r,dd v,int k){
28      dd m=(l+r)*.5;
```

```
29      dd v1=area(l,m),v2=area(m,r);
30      if(sig(v1+v2-v)==0&&k>4)rt v;//避免空白区域的影响
31      rt dfs(l,m,v1,k+1)+dfs(m,r,v2,k+1);
32  }
33  int main(){
34      while(~scanf("%d",&n)){
35          for(k=0;k<n;k++){
36              s[k].get();
37              printf("%.8lf\n",dfs(s[k].a.x,s[k].b.x,area(s[k].a.x,s[k].b.x),0)
                    /s[k].A2());
38          }
39      }
40      return 0;
41  }
```

2.6.3 圆面积并，常数优化 Simpson

```
 1  #include <cstdio>
 2  #include <algorithm>
 3  #include <cmath>
 4  using namespace std;
 5  #define dd double
 6  #define rt return
 7  #define ct const
 8  #define cpt ct pt&
 9  ct dd eps=1e-8;
10  inline int sig(dd x){rt (x>eps)-(x<-eps);}
11  struct cc{
12      dd x,y,r;
13      void get(){scanf("%lf%lf%lf",&x,&y,&r);}
14      bool in(ct cc &c){
15          rt sqrt((x-c.x)*(x-c.x)+(y-c.y)*(y-c.y))<r-c.r+eps;
16      }
17  }c[1010];
18  #define sim(l,r,sl,sr,sm) ((sl+sr+4.0*sm)*(r-l)/6.0)
19  dd eve[2010];
20  int id[2010];
21  bool cmp(int a,int b){rt eve[a]<eve[b];}
22  struct merge_cc{
23      dd area;
24      cc *c;
25      int n;
26      merge_cc();
27      merge_cc(cc _c[],int _n){
28          c=_c,n=_n;
29          area=0;
30          for(int i=0;i<n;i++)
31              for(int j=0;j<n;j++)
32                  if(i>=0&&i!=j&&c[j].in(c[i]))
33                      c[i--]=c[--n];
34          dd l=c[0].x-c[0].r,r=c[0].x+c[0].r;
35          for(int i=1;i<n;i++){
36              l=min(c[i].x-c[i].r,l);
37              r=max(c[i].x+c[i].r,r);
38          }
39          dd m=(l+r)*.5;
40          dd sl=calc(l),sr=calc(r),sm=calc(m);
41          area=dfs(l,r,m,sl,sr,sm,sim(l,r,sl,sr,sm),0);
42      }
43      dd dfs(dd l,dd r,dd m,dd sl,dd sr,dd sm,dd v,int k){
44          dd lm=(l+m)*.5,rm=(m+r)*.5;
45          dd slm=calc(lm),srm=calc(rm);
46          dd vl=sim(l,m,sl,sm,slm),vr=sim(m,r,sm,sr,srm);
47          if(sig(vl+vr-v)==0&&k>4)rt v;
48          rt dfs(l,m,lm,sl,sm,slm,vl,k+1)+dfs(m,r,rm,sm,sr,srm,vr,k+1);
49      }
```

```
50      dd calc(dd x){
51          int e=0;
52          for(int i=0;i<n;i++){
53              dd t=c[i].r*c[i].r-(x-c[i].x)*(x-c[i].x);
54              if(t<eps)continue;
55              t=sqrt(t);
56              eve[e++]=c[i].y-t;
57              eve[e++]=c[i].y+t;
58          }
59          for(int i=0;i<e;i++)id[i]=i;
60          sort(id,id+e,cmp);
61          int cnt=0;
62          dd s=0;
63          for(int i=0;i<e;i++){
64              if(cnt)s+=eve[id[i]]-eve[id[i-1]];
65              if(id[i]&1)cnt--;
66              else cnt++;
67          }
68          rt s;
69      }
70  };
71  int main(){
72      int n;
73      while(~scanf("%d",&n)){
74          for(int i=0;i<n;i++){
75              c[i].get();
76              if(c[i].r<eps)i--,n--;
77          }
78          printf("%.3lf\n",merge_cc(c,n).area);
79      }
80      rt 0;
81  }
```

2.6.4 5 个球和四面体的体积并

在三维问题里面，精度有些问题，速度不快。

```
1  #include <cstdio>
2  #include <algorithm>
3  #include <cmath>
4  using namespace std;
5  #define max(a,b) ((a)>(b)?(a):(b))
6  #define min(a,b) ((a)<(b)?(a):(b))
7  #define dd double
8  #define rt return
9  #define ct const
10 #define cpt ct pt&
11 #define op operator
12 ct dd eps=1e-6;
13 ct dd pi=acos(-1.0);
14 inline int sig(dd x){rt (x>eps)-(x<-eps);}
15 #define sim(l,r,sl,sr,sm) ((sl+sr+4.0*sm)*(r-l)/6.0)
16 dd eve[100];
17 int id[100];
18 struct pt{
19     dd x,y,z;
20     void get(){scanf("%lf%lf%lf",&x,&y,&z);}
21     void out(){printf("%lf␣%lf␣%lf\n",x,y,z);}
22     bool get_pt(cpt a,cpt b,dd zz){
23         if(sig(a.z-b.z)==0||zz>a.z+eps&&zz>b.z+eps||zz<a.z-eps&&zz<b.z-eps)rt
               0;
24         x=(zz-a.z)*(b.x-a.x)/(b.z-a.z)+a.x;
25         y=(zz-a.z)*(b.y-a.y)/(b.z-a.z)+a.y;
26         z=0;
27         rt 1;
28     }
29
30 };
31 struct poly{
```

```
32      pt p[4];
33      void get(){
34          for(int i=0;i<4;i++)p[i].get();
35      }
36      bool calc(dd x,dd z,dd &a,dd &b){
37          a=1e100,b=-1e100;
38          int n=0;
39          pt tp[5];
40          for(int i=0;i<4;i++)
41              for(int j=i+1;j<4;j++)
42                  if(tp[n].get_pt(p[i],p[j],z))n++;
43          if(n==0)rt 0;
44          for(int i=0;i<n;i++)
45              for(int j=i+1;j<n;j++){
46                  if(sig(tp[i].x-tp[j].x)==0||x<tp[i].x-eps&&x<tp[j].x-eps||x>
                       tp[i].x+eps&&x>tp[j].x+eps)continue;
47                  dd y=(x-tp[i].x)*(tp[j].y-tp[i].y)/(tp[j].x-tp[i].x)+tp[i].y;
48                  a=min(a,y);
49                  b=max(b,y);
50              }
51          rt a<b;
52      }
53  }g[10];
54  struct cc{
55      pt o;
56      dd r;
57      void get(){o.get(),scanf("%lf",&r);}
58      bool calc(dd x,dd z,dd &a,dd &b){
59          dd r2=r*r-(o.z-z)*(o.z-z);
60          if(r2<eps*eps)rt 0;
61          r2-=(o.x-x)*(o.x-x);
62          if(r2<eps*eps)rt 0;
63          r2=sqrt(r2);
64          a=o.y-r2,b=o.y+r2;
65          rt 1;
66      }
67  }c[10];
68  int n,m;
69  bool cmp(int a,int b){rt eve[a]<eve[b];}
70  dd calc(dd x,dd z){
71      int e=0;
72      for(int i=0;i<n;i++)
73          if(g[i].calc(x,z,eve[e],eve[e+1]))e+=2;
74      for(int i=0;i<m;i++)
75          if(c[i].calc(x,z,eve[e],eve[e+1]))e+=2;
76      for(int i=0;i<e;i++)id[i]=i;
77      sort(id,id+e,cmp);
78      int cnt=0;
79      dd s=0;
80      for(int i=0;i<e;i++){
81          if(cnt)s+=eve[id[i]]-eve[id[i-1]];
82          if(id[i]&1)cnt--;
83          else cnt++;
84      }
85      rt s;
86  }
87  dd dfs(dd l,dd r,dd m,dd sl,dd sr,dd sm,dd v,dd z,int k){
88      dd lm=(l+m)*.5,rm=(m+r)*.5;
89      dd slm=calc(lm,z),srm=calc(rm,z);
90      dd vl=sim(l,m,sl,sm,slm),vr=sim(m,r,sm,sr,srm);
91      if(sig(vl+vr-v)==0&&k>4)rt v;
92      rt dfs(l,m,lm,sl,sm,slm,vl,z,k+1)+dfs(m,r,rm,sm,sr,srm,vr,z,k+1);
93  }
94  dd area(dd z){
95      dd l=-8,r=8,m=0;//边界x
96      dd sl=calc(l,z),sr=calc(r,z),sm=calc(m,z);
97      rt dfs(l,r,m,sl,sr,sm,sim(l,r,sl,sr,sm),z,0);
```

```
 98 }
 99 dd dfs(dd l,dd r,dd m,dd sl,dd sr,dd sm,dd v,int k){
100     dd lm=(l+m)*.5,rm=(m+r)*.5;
101     dd slm=area(lm),srm=area(rm);
102     dd vl=sim(l,m,sl,sm,slm),vr=sim(m,r,sm,sr,srm);
103     if(sig(vl+vr-v)==0&&k>4)rt v;
104     rt dfs(l,m,lm,sl,sm,slm,vl,k+1)+dfs(m,r,rm,sm,sr,srm,vr,k+1);
105 }
106 dd vol(){
107     dd l=-8,r=8,m=0;//边界z
108     dd sl=area(l),sr=area(r),sm=area(m);
109     rt dfs(l,r,m,sl,sr,sm,sim(l,r,sl,sr,sm),0);
110 }
111 int main(){
112     while(scanf("%d%d",&n,&m),n||m){
113         for(int i=0;i<n;i++)g[i].get();
114         for(int i=0;i<m;i++)c[i].get();
115         printf("%.3lf\n",vol());
116     }
117     rt 0;
118 }
```

2.6.5 多边形面积并

和确定做法有一定误差，速度也不是很快。

```
 1 #include <cstdio>
 2 #include <algorithm>
 3 #include <cmath>
 4 using namespace std;
 5 #define max(a,b) ((a)>(b)?(a):(b))
 6 #define min(a,b) ((a)<(b)?(a):(b))
 7 #define dd double
 8 #define rt return
 9 #define ct const
10 #define cpt ct pt&
11 #define op operator
12 ct dd eps=1e-8;
13 inline int sig(dd x){rt (x>eps)-(x<-eps);}
14 #define sim(l,r,sl,sr) ((sl+sr)*(r-l)/2.0)
15 dd eve[220];
16 int id[220];
17 struct pt{
18     dd x,y;
19     void get(){scanf("%lf%lf",&x,&y);}
20 }p[110][4];
21 int n;
22 bool cmp(int a,int b){rt eve[a]+eps<eve[b];}
23 bool SS(pt p[],dd x,dd &a,dd &b){
24     a=1e100,b=-1e100;
25     for(int i=0;i<3;i++){
26         if(p[i].x==p[i+1].x||x<p[i].x-eps&&x<p[i+1].x-eps||x>p[i].x+eps&&x>p[
              i+1].x+eps)continue;
27         dd y=(x-p[i].x)*(p[i+1].y-p[i].y)/(p[i+1].x-p[i].x)+p[i].y;
28         a=min(a,y);
29         b=max(b,y);
30     }
31     rt a<b+eps;
32 }
33 dd calc(dd x){
34     int e=0;
35     for(int i=0;i<n;i++)
36         if(SS(p[i],x,eve[e],eve[e+1]))e+=2;
37     for(int i=0;i<e;i++)id[i]=i;
38     sort(id,id+e,cmp);
39     int cnt=0;
40     dd s=0;
```

```
    for(int i=0;i<e;i++){
        if(cnt)s+=eve[id[i]]-eve[id[i-1]];
        if(id[i]&1)cnt--;
        else cnt++;
    }
    rt s;
}
dd dfs(dd l,dd r,dd sl,dd sr,dd v,int k){
    dd m=(l+r)*.5,sm=calc(m);
    dd vl=sim(l,m,sl,sm),vr=sim(m,r,sm,sr);
    if(sig(vl+vr-v)==0&&k>9)rt v;
    rt dfs(l,m,sl,sm,vl,k+1)+dfs(m,r,sm,sr,vr,k+1);
}
dd area(dd l,dd r){
    if(l>r)rt 0;
    dd sl=calc(l),sr=calc(r);
    rt dfs(l,r,sl,sr,sim(l,r,sl,sr),0);
}
int main(){
    int T,cas=1;
    scanf("%d",&T);
    while(T--){
        scanf("%d",&n);
        dd l=1e100,r=-1e100;
        for(int i=0;i<n;i++){
            for(int j=0;j<3;j++){
                p[i][j].get();
                l=min(l,p[i][j].x);
                r=max(r,p[i][j].x);
            }
            p[i][3]=p[i][0];
        }
        printf("Case␣%d:␣%.3lf",cas++,area(l,r));
    }
    rt 0;
}
```

2.7 计算几何之浮点数

```
extern float frexp(float x, int *exp);
//把浮点数分解成尾数和指数x.
extern float hypot(float x, float y);
//对于给定的直角三角形的两个直角边，求其斜边的长度。
extern float modf(float num, float *i);
//将浮点数分解成整数部分和小数部分。num
extern float sinh(float x);
//计算（弧度表示）的双曲正弦值。xsinh(x)=(e^x-e^(-x))。/2
extern float cosh(float x);
//求的双曲余弦值x,cosh(x)=(e^x+e^(-x))/2
extern float tanh(float x);
//求的双曲正切值x,tanh(x)=(e^x-e^(-x))/(e^2+e^(-x))
```

2.7.1 浮点数修正

```
inline dd fix(dd x,int t=0){
    if(t==0)rt x>-eps?x+eps:x-eps; //output for double
    if(t==1){ //for sqrt & log
        rt x+(x<eps?eps:0);
    }
    if(t==2){ //for sin & cos
        if(x>1)rt 1;
        if(x<-1)rt -1;
        rt x;
    }
}
```

2.7.2 读入优化

万不得已时用，计算规则和 scanf 有出入，可能会出错。

```
inline void in(dd &x){
    x=0;
    char c;
    while(c=getchar(),(c<'0'||c>'9')&&c!='-'&&c!='.');
    dd f=1;
    if(c!='.'){
        if(c=='-')f=-1;
        else x=c-'0';
        while(c=getchar(),c>='0'&&c<='9')x=x*10+c-'0';
    }
    if(c=='.'){
        dd t=1;
        while(c=getchar(),c>='0'&&c<='9')t*=0.1,x+=t*(c-'0');
    }
    x=f*x;
}
```

2.8 点的基本操作

```
//椭球体积 sqrt(a*b*sqrt(c))*pi*160.0
//e^ix=cosx+isinx
#include <stdio.h>
#include <algorithm>
#include <math.h>
#include <time.h>
using namespace std;
#define cpt ct pt&
#define op operator
#define dd double
#define rt return
#define ct const
ct dd eps=1e-8;
ct dd pi=acos(-1.0);
inline int sig(ct dd &x){rt (x>eps)-(x<-eps);}
struct pt{
    dd x,y;
    pt(){}
    pt(dd x,dd y):x(x),y(y){}
    void get(){scanf("%lf%lf",&x,&y);}
    void out(){printf("%lf %lf\n",x,y);}
    pt op+(cpt p)ct{rt pt(x+p.x,y+p.y);}
    pt op-(cpt p)ct{rt pt(x-p.x,y-p.y);}
    pt op*(dd v)ct{rt pt(x*v,y*v);}
    dd op*(cpt p)ct{rt x*p.x+y*p.y;}
    dd op^(cpt p)ct{rt x*p.y-y*p.x;}
    bool op<(cpt p)ct{rt y+eps<p.y||y<p.y+eps&&x+eps<p.x;}
    dd X(cpt a,cpt b)ct{rt a-*this^b-*this;}
    dd O(cpt a,cpt b)ct{rt (a-*this)*(b-*this);}
    dd abs2()ct{rt x*x+y*y;}
    dd abs()ct{rt sqrt(x*x+y*y);}
    bool op==(cpt p)ct{rt sig(x-p.x)==0&&sig(y-p.y)==0;}
    pt TR()ct{rt pt(-y,x);}
    pt norm()ct{rt pt(y,-x)*(1/abs());}/*顺时针单位向量*/
    pt(cpt a,cpt b,cpt c,cpt d){
        dd t=(d-c^a-c)/(b-a^d-c);
        *this=a*(1-t)+b*t;
    }
    pt R(pt o,dd v)ct{
        dd s=sin(v),c=cos(v);
        pt t=*this-o;
        pt p=pt(t.x*c-t.y*s,t.x*s+t.y*c);
        rt p+o;
```

```
    }
    pt Z(pt a,pt b,dd n1,dd n2){
        dd v=atan2(X(a,b),O(a,b));
        dd v1=asin(sin(pi/2-v)*n1/n2);
        rt b.R(*this,sig(v)*(pi/2+v1));
    }
    pt F(pt a,pt b){
        dd v=atan2(X(a,b),O(a,b));
        rt b.R(*this,pi+v);
    }
    pt pro(cpt a,cpt b){
        rt a+(b-a)*(a.O(b,*this)/a.O(b,b));
    }
    pt D(cpt a,cpt b){
        rt pro(a,b)*2-*this;
    }
    bool on(cpt a,cpt b){
        rt sig(X(a,b))==0&&sig(O(a,b))<=0;
    }
    bool SS(cpt a,cpt b,cpt c,cpt d){
        dd u=(b-a^d-c);
        if(!sig(u))rt 0;
        dd t=c.X(d,a)/u,s=a.X(c,b)/u;
        if(t<0||t>1||s<0||s>1)rt 0;/*eps !!!*/
        *this=a*(1-t)+b*t;
        rt 1;
    }
    dd dis_line(cpt a,cpt b)ct{
        rt fabs(X(a,b))/(a-b).abs();
    }
    dd dis_seg(cpt p,cpt q)ct{
        if(sig(p.O(*this,q)<=0))rt (*this-p).abs();
        if(sig(q.O(*this,p)<=0))rt (*this-q).abs();
        rt dis_line(p,q);
    }
};
dd angle(cpt u,cpt v){/*vector*/
    rt atan2(u^v,u*v);
}
dd seg_seg(cpt a,cpt b,cpt c,cpt d){
    rt min(min(a.dis_seg(c,d),b.dis_seg(c,d)),min(c.dis_seg(a,b),d.dis_seg(a,
       b)));
}
```

2.9 多边形基本操作,p[n]=p[0]

2.9.1 分式线性变换

二维变换的终极利器，不用三角函数

```
void RST(cpt a,cpt b,cpt c,cpt d,pt ch[]){//c-d-->a-b
        pt ab=b-a,cd=d-c;
        dd s=cd*cd,u=(ab*cd)/s,v=(cd^ab)/s;
        for(int i=0;i<n;i++){
                dd x=ch[i].x-c.x,y=ch[i].y-c.y;
                p[i]=pt(u*x-v*y,v*x+u*y)+a;
        }
}
```

2.9.2 半平面交

a->b 的顺时针, 平移 t=(b-a).norm()*r,a=a+t,b=b+t; 逆时针 a=a-t,b=b-t;

```
void cut(cpt a,cpt b,pt ch[],int &m){/*=*/
        int n=0;
        for(int i=0;i<m;i++){
                int u=sig(a.X(b,ch[i])),v=sig(a.X(b,ch[i+1]));
                if(u>=0)p[n++]=ch[i];/*>= 逆时针*/
```

```
6                   if(u*v<0)p[n++]=pt(a,b,ch[i],ch[i+1]);/*< 逆时针*/
7           }
8           m=n;
9           for(int i=0;i<m;i++)ch[i]=p[i];
10          ch[m]=ch[0];
11  }
12  //用于成员函数
13  bool cut(cpt a,cpt b,pt ch[],int m){/*=*/
14          n=0;
15          for(int i=0;i<m;i++){
16                  int u=sig(a.X(b,ch[i])),v=sig(a.X(b,ch[i+1]));
17                  if(u>=0)p[n++]=ch[i];/*>= 逆时针*/
18                  if(u*v<0)p[n++]=pt(a,b,ch[i],ch[i+1]);/*< 逆时针*/
19          }
20          p[n]=p[0];
21          rt n!=0;
22  }
```

2.9.3 凸包，水平序

```
1   void convex1(pt ch[],int m){/*不要边上的点*/
2           n=0;
3           sort(ch,ch+m);
4           for(int i=0;i<m;i++){
5                   while(n>1&&sig(p[n-2].X(p[n-1],ch[i]))<=0)n--;
6                   p[n++]=ch[i];
7           }
8           int k=n;
9           for(int i=m-2;i>=0;i--){
10                  while(n>k&&sig(p[n-2].X(p[n-1],ch[i]))<=0)n--;
11                          p[n++]=ch[i];
12          }
13          if(m>1)n--;
14          p[n]=p[0];
15  }
16  void convex2(pt ch[],int m){/*要边上的点*/
17          n=0;
18          sort(ch,ch+m);
19          for(int i=0;i<m;i++){
20                  while(n>1&&sig(p[n-2].X(p[n-1],ch[i]))<0)n--;
21                  p[n++]=ch[i];
22          }
23          if(n==m)rt;
24          int k=n;
25          for(int i=m-2;i>=0;i--){
26                  while(n>k&&sig(p[n-2].X(p[n-1],ch[i]))<0)n--;
27                  p[n++]=ch[i];
28          }
29          if(m>0)n--;
30          p[n]=p[0];
31  }
```

2.9.4 凸包，graham

```
1   pt pk;
2   bool g_c(cpt a,cpt b){
3       int u=sig(pk.X(a,b));
4       rt u>0||u==0&&sig(pk.O(a,a)-pk.O(b,b))<0;
5   }
6   void convex(){/*不要边上的点*/
7           int k=0;
8           for(int i=1;i<n;i++)
9                   if(p[i].y+eps<p[k].y||p[i].y<p[k].y+eps&&p[i].x+eps<p[k].x)k=
                        i;
10          pk=p[k];
11          sort(p,p+n,g_c);
12          if(n<=1){rt;}
```

```
13          if(sig(p[0].X(p[1],p[n-1]))==0){
14                  p[1]=p[n-1];n=2;rt;
15          }
16          p[n++]=p[0];
17          int m=n;
18          n=1;
19          for(int i=2;i<m;i++){
20                  while(n>0&&sig(p[n-1].X(p[n],p[i]))<=0)n--;
21                  p[++n]=p[i];
22          }
23          p[n]=p[0];
24  }
25  void convex(){/*要边上的点*/
26          int k=0;
27          for(int i=1;i<n;i++)
28                  if(p[i].y+eps<p[k].y||p[i].y<p[k].y+eps&&p[i].x+eps<p[k].x)k=
                        i;
29          pk=p[k];
30          sort(p,p+n,g_c);
31          if(n>0&&sig(p[0].X(p[1],p[n-1]))){
32                  int t=n-1;
33                  while(sig(p[0].X(p[n-1],p[t]))==0)t--;
34                  reverse(p+t+1,p+n);
35          }
36          int m=n;
37          n=0;
38          for(int i=0;i<m;i++){
39                  while(n>=2&&sig(p[n-2].X(p[n-1],p[i]))<0)n--;
40                  p[n++]=p[i];
41          }
42          p[n]=p[0];
43  }
```

2.9.5 点在多边形内

```
 1  bool inside_me(cpt c){//on, ?o(n)
 2          bool in=0;
 3          for(int i=0;i<n;i++){
 4                  pt a=p[i],b=p[i+1];
 5                  if(((b.x<c.x)^(a.x<c.x))&&((b.y-a.y)*abs(c.x-a.x)<(c.y-a.y)*
                        abs(b.x-a.x)))in=!in;
 6          }
 7          rt in;
 8  }
 9  bool inside_me(cpt c){//on?,log(n)
10          if(n<3)rt c.on(p[0],p[1]);
11          int l=1,r=n-1;
12          if(p[0].X(p[r],c)<=0&&p[0].X(p[l],c)>=0){
13                  while(l+1<r){
14                          int m=(l+r)>>1;
15                          if(p[0].X(p[m],c)>=0)l=m;
16                          else r=m;
17                  }
18          }else rt 0;
19          rt p[l].X(p[l+1],c)>=0;
20  }
```

2.9.6 面积，重心等

```
 1  dd _area(){/*-+-+-+*/
 2          dd s=0;p[n]=p[0];
 3          for(int i=0;i<n;i++)s+=(p[i]^p[i+1]);
 4          rt s*0.5;
 5  }
 6  pt heart(){
 7          pt t(0,0);
 8          dd sum=0,s;
 9          for(int i=0;i<n;i++,sum+=s){
```

```
10                  s=(p[i]^p[i+1]);
11                  t=t+(p[i]+p[i+1])*s;
12          }
13          rt t*(1/(3*sum));
14  }
15  dd D2(){//p[n]=p[0];max(p[i+1].O(p[j+1],p[j+1]);
16          dd s=0;
17          for(int i=0,j=1;i<n;i++){
18                  while(p[i].X(p[i+1],p[j])<p[i].X(p[i+1],p[j+1]))j=(j+1)%n;
19                  s=max(s,p[i].O(p[j],p[j]));
20          }
21          rt s;
22  }
23  dd rotcal(poly &b) {
24          dd s=1e100;
25          for(int i=0,j=0,t;i<n;i++){
26                  while(t=sig(p[i].X(p[i+1],b.p[j+1])-p[i].X(p[i+1],b.p[j])),t
                       >0)j=(j+1)%b.n;
27                  if(t)s=min(s,b.p[j].dis_seg(p[i],p[i+1]));
28                  else s=min(s,seg_seg(p[i],p[i+1],b.p[j],b.p[j+1]));
29          }
30          rt s;
31  }
```

2.9.7 区间操作

不会扫描线的时候写的，没什么用

```
 1  dd length(){
 2          dd l=0,r=0,sum=0;
 3          for(int i=0;i<n;i++)
 4                  if(sig(p[i].x-r)<=0)r=max(r,p[i].y);
 5                  else{
 6                          sum+=(r-l);
 7                          l=p[i].x,r=p[i].y;
 8                  }
 9          rt sum+(r-l);
10  }
11  void merge(){
12          if(n==0)rt;
13          sort(p,p+n);
14          int k=n;n=0;
15          pt t=p[0];
16          for(int i=1;i<k;i++)
17                  if(sig(p[i].x-t.y)<=0)t.y=max(t.y,p[i].y);
18                  else {
19                          p[n++]=t;
20                          t=p[i];
21                  }
22          p[n++]=t;
23  }
24  void join(poly &a,poly &b){
25          n=0;
26          for(int i=0,j=0;i<a.n&&j<b.n;){
27                  if(a.p[i].y<b.p[j].x+eps)i++;
28                  else if(b.p[j].y<a.p[i].x+eps)j++;
29                  else if(a.p[i].y+eps<b.p[j].y)p[n++]=pt(max(a.p[i].x,b.p[j].x
                       ),a.p[i].y),i++;
30                  else if(b.p[j].y+eps<a.p[i].y)p[n++]=pt(max(a.p[i].x,b.p[j].x
                       ),b.p[j].y),j++;
31                  else p[n++]=pt(max(a.p[i].x,b.p[j].x),b.p[j].y),j++,i++;
32          }
33          if(a.n==0)*this=b;
34  }
```

2.10 圆的基本操作

```
struct cc{
    dd r;
    pt o;
    void get(){o.get(),scanf("%lf",&r);}
    cc(){}
    cc(cpt a,cpt b,cpt c){//外接圆保证不共线,
        pt u=b-a,v=c-a;
        o=a+pt(u,u+u.TR(),v,v+v.TR())*.5;
        r=(a-o).abs();
    }
    cc(cpt a,cpt b,cpt c){//内切圆
                dd A=(b-c).abs(),B=(a-c).abs(),C=(a-b).abs();
                o=(a*A+b*B+c*C)/(A+b+C)
                r=(a-o).abs();
    }
    cc(int n,int k){//random_shuffle(p,p+n);cc c(n,0);
        if(k==0)r=-1;
        else if(k==1)o=tr[0],r=0;
        else if(k==2)o=(tr[0]+tr[1])*.5,r=(tr[0]-o).abs();
        else if(k==3){
            *this=cc(tr[0],tr[1],tr[2]);
            rt;
        }
        for(int i=0;i<n;i++)
            if((p[i]-o).abs()>r+eps){
                tr[k]=p[i];
                *this=cc(i,k+1);
                pt t=p[i];
                for(int j=i;j>0;j--)p[j]=p[j-1];
                p[0] = t;
            }
    }
    //用于求交点，不用于判相交面积，情况过多，比如内接多边形。
    void inter_me(cpt a,cpt b,pt p[],int &n){
        n=0;
        pt ab=b-a,oa=a-o;
        dd A=ab*ab,B=ab*oa,C=oa*oa-r*r,D=B*B-A*C;
        if(D>eps){
            D=sqrt(D);
            dd t1=(-B-D)/A;
            if(t1>0&&t1<1)p[n++]=a*(1-t1)+b*t1;
            dd t2=(-B+D)/A;
            if(t2>0&&t2<1)p[n++]=a*(1-t2)+b*t2;
        }
    }
    void inter_me(cc &t,pt p[],int &n){
        n=0;
        pt e=(t.o-o);
        dd d=e.abs();
        if(sig(d)){
            dd x=(d*d-t.r*t.r+r*r)/(d+d),h=r*r-x*x;
            if(sig(h)>=0){
                                h=sqrt(h);
                e=e*(1/d);
                pt no=e.TR();
                p[n++]=o+e*x+no*h;
                if(sig(h))p[n++]=o+e*x-no*h;
            }
        }
    }
    void c_c(ct cc& c){//圆覆盖了某圆的弧度起点与重点
        pt dir=c.o-o;
        dd d=dir.abs();
        if(r+c.r<d+eps)rt;//相离，不处理全包含
        if(c.r>r-eps&&c.r-r>d+eps){//包含，处理全包含
            eve[e++]=-pi,eve[e++]=pi;
            rt;
```

```
68          }
69          dd v=atan2(dir.y,dir.x);//(-pi,pi]
70          dd add=acos(((d*d+r*r-c.r*c.r)/d)/(r+r));//[0,pi]
71          eve[e++]=v-add,eve[e++]=v+add;//(-2pi,2pi]&&a<b&&(b-a)[0,pi]
72      }
73      void c_c(cpt a,cpt b){//直线覆盖顺时针方向的弧ab
74          dd d=(o.X(a,b)/(b-a).abs());
75  //        if(d>r-eps||d<-r+eps)rt不处理全包含;//
76          if(d>r-eps)rt;//离
77          if(d<-r+eps){//反向离，处理全包含
78              eve[e++]=-pi,eve[e++]=pi;
79              rt;
80          }
81          pt dir=(a-b).TR();
82          dd v=atan2(dir.y,dir.x),add=acos(d/r);
83          eve[e++]=v-add,eve[e++]=v+add;
84      }
85      dd A2(cpt a,cpt b){//圆与三角形交注意三角函数的参数,
86          dd A=(b-o).abs(),B=(a-o).abs(),C=(b-a).abs(),s=o.X(a,b);
87          if(A<r&&B<r)rt s;
88          if(A<r&&B>=r){
89              dd x=(b.O(a,o)+sqrt(r*r*C*C-s*s))/C;
90              return asin(s*(1-x/C)/r/B)*r*r+s*x/C;
91          }
92          if(A>=r&&B<r){
93              dd y=(a.O(b,o)+sqrt(r*r*C*C-s*s))/C;
94              rt asin(s*(1-y/C)/r/A)*r*r+s*y/C;
95          }
96          if(fabs(s)>=r*C||a.O(b,o)<=0||b.O(a,o)<=0){
97              dd si=asin(fix(s/A/B,2))*r*r;
98              if(o.O(a,b)<0){
99                  dd co=acos(-1.0)*r*r;
100                                  rt -si+(s<0?-co:co);
101             }
102             rt si;
103         }
104         dd x=(b.O(a,o)+sqrt(r*r*C*C-s*s))/C,y=(a.O(b,o)+sqrt(r*r*C*C-s*s))/C;
105         rt asin(s*(1-x/C)/r/B)*r*r+s*x/C+asin(s*(1-y/C)/r/A)*r*r+s*y/C-s;
106     }
107 };
108 bool make_tri(dd a,dd b,dd c,pt p[]){//三边长确定三角形
109     if(a>b)swap(a,b);
110     if(b>c)swap(b,c);
111     if(a>b)swap(a,b);
112     if(a+b<=c)return 0;
113     p[0]=pt(0,0);
114     p[1]=pt(c,0);
115     p[2].x=0.5*(c*c-a*a+b*b)/c;
116     p[2].y=sqrt(b*b-p[2].x*p[2].x);
117     return 1;
118 }
119 dd gong(dd r,dd x){//半径，高可以为负(())
120     rt r*r*acos(x/r)-sqrt(r*r-x*x)*x;
121 }
122 dd c_c(dd r,dd R,dd c){//为圆心距c
123     if(r>R)swap(r,R);
124     if(r+R<c+eps)rt 0;
125     if(R-r+eps>c)rt r*r*pi;
126     dd t=(R*R-r*r)/c;
127     dd x=(c-t)*0.5,y=(c+t)*0.5;
128     rt gong(r,x)+gong(R,y);
129 }
```

2.10.1 最近点对

```
1  pt p[N];
2  int on,o[N],n;
3  bool c_x(cpt a,cpt b){rt a.x+eps<b.x;}
```

```
bool c_y(ct int &a,ct int &b){rt p[a].y+eps<p[b].y;}
dd c_pair(int l,int r){
    dd ret=1e100;
    if(l>=r)rt ret;
    int m=(l+r)>>1,i;
    for(i=m;i>=l&&!sig(p[i].x-p[m].x);i--);
    ret=c_pair(l,i);
    for(i=m;i<=r&&!sig(p[i].x-p[m].x);i++);
    ret=min(ret,c_pair(i,r));
    on=0;
    for(i=m;i>=l&&sig(p[m].x-p[i].x-ret)<=0;i--)o[++on]=i;
    for(i=m+1;i<=r&&sig(p[i].x-p[m].x-ret)<=0;i++)o[++on]=i;
    sort(o+1,o+on+1,c_y);
    for(i=1;i<=on;i++)
        for(int j=1;j<9&&i+j<=on;j++)
          ret=min(ret,(p[o[i]]-p[o[i+j]]).abs());
    rt ret;
}
dd c_pair(){
  sort(p,p+n,c_x);
  rt c_pair(0,n-1);
}
```

2.11 pick 定理

面积 = 内部点 + 边界点/2-1

```
int gcd(int a,int b){
  rt b?gcd(b,a%b):a;
}
int gird_onedge(){
  int cnt=0;p[n]=p[0];
  for(int i=0;i<n;i++)
      cnt+=gcd(abs(p[i].x-p[i+1].x),abs(p[i].y-p[i+1].y));
  rt cnt;
}
int grid_inside(){
  int cnt=0;p[n]=p[0];p[n+1]=p[1];
  for(int i=0;i<n;i++)
      cnt+=p[i+1].y*(p[i].x-p[i+2].x);
  rt (abs(cnt)-gird_onedge())/2+1;
}
```

2.12 三维几何基础

浮点乘法不重载不报错!!
注意初始化 N
I'm not sure about this.
椭球体积 sqrt(a*b*sqrt(c))*pi*160.0
参数 v 多是向量
法向量注意可能为 0
台体体积：上底面 S1，下底面 S2，中间截面 S0，高 H，
V=(S1+sqrt(S1*S2)+S2)*H/3,2sqrt(S0)=sqrt(S1)+sqrt(S2);
拟柱体体积：V=(S1+4*S0+S2)*H/6;

2.12.1 基本操作

```
struct pt{
    dd x,y,z;
    pt(dd x=0,dd y=0,dd z=0):x(x),y(y),z(z){}
    void get(){scanf("%lf%lf%lf",&x,&y,&z);}
    void out(){printf("%lf␣%lf␣%lf\n",x,y,z);}
    pt op+(cpt p)ct{rt pt(x+p.x,y+p.y,z+p.z);}
    pt op-(cpt p)ct{rt pt(x-p.x,y-p.y,z-p.z);}
    pt op*(dd v)ct{rt pt(x*v,y*v,z*v);}
```

```
 9      dd op*(cpt p)ct{rt x*p.x+y*p.y+z*p.z;}
10      pt op^(cpt p)ct{rt pt(y*p.z-z*p.y,z*p.x-x*p.z,x*p.y-y*p.x);}
11      pt X(cpt a,cpt b)ct{rt a-*this^b-*this;}
12      dd O(cpt a,cpt b)ct{rt (a-*this)*(b-*this);}
13      bool op<(cpt p)ct{rt x+eps<p.x||x<p.x+eps&&(y+eps<p.y||y<p.y+eps&&z+eps<p
           .z);}
14      bool op==(cpt p)ct{rt sig(x-p.x)==0&&sig(y-p.y)==0&&sig(z-p.z)==0;}
15      dd V6(cpt a,cpt b,cpt c)ct{rt a.X(b,c)*(*this-a);}
16      dd V6(cpt b,cpt c)ct{rt (b^c)*(*this);}
17      dd A2(cpt b,cpt c)ct{rt X(b,c).abs();}
18      int side(cpt a,cpt b,cpt c)ct{rt sig(V6(a,b,c));}
19      dd abs2(){rt x*x+y*y+z*z;}
20      dd abs(){rt sqrt(x*x+y*y+z*z);}
21      dd abs(cpt v){rt sig(*this*v)*abs();}//相对有向
22      dd dis_line(cpt a,cpt b){rt X(a,b).abs()/(a-b).abs();}
23      bool on(cpt a,cpt b)ct{
24          rt X(a,b).abs()<eps&&O(a,b)<eps;
25      }
26      //绕过原点的单位向量旋转角度后得到的点vv注意旋转轴构造的特殊情况,
27      void R(dd v,pt r[]){
28          *this=(*this)*(1/abs());
29          dd c=cos(v),s=sin(v);
30          r[0]=pt(c+(1-c)*x*x,  (1-c)*x*y-s*z, (1-c)*x*z+s*y);
31          r[1]=pt((1-c)*y*x+s*z, c+(1-c)*y*y,  (1-c)*y*z-s*x);
32          r[2]=pt((1-c)*z*x-s*y, (1-c)*z*y+s*x, c+(1-c)*z*z);
33      }
34      pt(cpt a,cpt b,cpt c,cpt d){//一定有交
35          pt to=(b-a^d-c);
36          dd u=to.abs();
37          dd t=c.X(d,a).abs(to)/u;
38          *this=a*(1-t)+b*t;
39      }
40      bool SS(cpt a,cpt b,cpt c,cpt d){
41          if(a.side(b,c,d))rt 0;//异面
42          pt to=(b-a^d-c);
43          dd u=to.abs();
44          if(!sig(u))rt 0;//平行
45          dd t=c.X(d,a).abs(to)/u,s=a.X(c,b).abs(to)/u;
46          *this=a*(1-t)+b*t;
47          rt t>=0&&t<=1&&s>=0&&s<=1;
48      }
49      pt TR(cpt b,cpt c){//(b,c)的逆时针垂直向量
50          rt (b+X(b,c)).X(c,b);
51      }
52      pt TL(cpt b,cpt c){
53          rt (b+X(b,c)).X(b,c);
54      }
55  };
56  bool T(pt a,pt b){
57      rt sig(a*b)==0;
58  }
59  bool H(pt a,pt b){
60      rt sig((a^b).abs2())==0;
61  }
62  dd line_line(cpt a,cpt b,cpt c,cpt d){/*必须是异面*/
63      pt to=(a-b)^(c-d);
64      rt fabs((a-c)*to)/to.abs();
65  }
66  dd sc(pt a,pt b){/*cos for line_line,sin for line_face*/
67      rt a*b/a.abs()/b.abs();
68  }
69  struct face{
70      pt a,b,c,d;
71      face(pt a,pt b,pt c):a(a),b(b),c(c),d(a.X(b,c)){}
72      bool on_me(cpt p){/*在三角形上包括边界,*/
73          rt sig(a.A2(b,c)-p.A2(a,b)-p.A2(b,c)-p.A2(c,a))==0;
74      }
```

```
75      bool in_me(cpt p){/*在三角形上不包括边界,*/
76          int v=sig(p.X(a,b)*d);
77          rt v==sig(p.X(b,c)*d)&&v==sig(p.X(c,a)*d);
78      }
79      dd dis_me(cpt p){
80          rt fabs(d*(p-a)/(d.abs()));
81      }
82      bool SF_inter(cpt p,cpt v,pt &x){
83          dd t=d*v;
84          if(sig(t)==0)rt 0;//平行
85          t=d*(a-p)/t;
86          if(t<0||t>1)rt 0;
87          x=p+v*t;
88          rt on_me(x);
89          rt in_me(x);
90      }
91      pt LP_inter(cpt p,pt v){/*不能平行*/
92          rt p+v*(d*(a-p)/(d*v));
93      }
94      bool PP_inter(face s,pt p,pt v){
95          v=d^s.d;
96          if(!sig(v.abs2()))rt 0;
97          p=T(a-b,s.d)?s.LP_inter(a,c-a):s.LP_inter(a,b-a);
98      }
99  };
100 pt get_T(cpt p,cpt a,cpt b){/*shunshizhen*/
101   rt p+((b-a)^p.X(a,b));
102 }
103 bool SB_inter(cpt A,cpt B,cpt C,dd r){
104   dd a=(B-C).abs(),b=(A-C).abs(),c=(A-B).abs();
105   rt acos((a*a+b*b-c*c)/(2*a*b))-acos(r/b)-acos(r/a)>=0;
106 }
107 struct cc{//bad
108     dd r;
109     pt o;
110     void get(){o.get(),scanf("%lf",&r);}
111     cc(){}
112     cc(cpt a,cpt b,cpt c){//外接圆
113         pt u=b-a,v=c-a;
114         o=a+pt(u,u+c.TR(a,b),v,v+b.TR(c,a))*.5;
115         r=(a-o).abs();
116     }
117     bool in_me(cpt p){//算边上
118         rt (p-o).abs()<r+eps;
119     }
120     void refresh(pt p[],int n){
121         pt t=p[n];
122         for(int i=n;i>0;i--)p[i]=p[i-1];
123         p[0]=t;
124     }
125     cc(pt p[],int n){
126         random_shuffle(p,p+n);//srand(time(0));
127         o=p[0],r=0;
128         for(int i=1;i<n;i++){
129             if(in_me(p[i]))continue;
130             o=p[i],r=0;
131             for(int j=0;j<i;j++){
132                 if(in_me(p[j]))continue;
133                 o=(p[i]+p[j])*.5,r=(p[j]-o).abs();
134                 for(int k=0;k<j;k++){
135                     if(in_me(p[k]))continue;
136                     *this=cc(p[i],p[j],p[k]);
137                     refresh(p,k);
138                 }
139                 refresh(p,j);
140             }
141             refresh(p,i);
```

```
        }
    }
};
```

2.12.2 三维凸包

```
struct convex{
  int n,to[N][N];
  pt *p;
  struct face{
      int a,b,c;
      bool ok;
  }f[N*10],add;
  int side(cpt t,face &f){
      rt sig(p[f.a].X(p[f.b],p[f.c])*(t-p[f.a]));
  }
  convex(){}
  void fixed(){/*注意共线共面*/
      add.a=0,add.b=1,add.c=2;
      int flag;
      for(flag=2;!sig(p[flag].X(p[0],p[1]).abs());flag++);
      swap(p[2],p[flag]);
      for(flag=3;!side(p[flag],add);flag++);
      swap(p[3],p[flag]);
  }
  convex(pt s[],int m){
      n=0;p=s;
      if(m<4)rt;
      fixed();
      for(int i=0;i<4;i++){
          add.a=(i+1)%4,add.b=(i+2)%4,add.c=(i+3)%4,add.ok=1;
          if(side(p[i],add)>0)swap(add.b,add.c);
          to[add.a][add.b]=to[add.b][add.c]=to[add.c][add.a]=n;
          f[n++]=add;
      }
      for(int i=4;i<m;i++){
          for(int j=0;j<n;j++){
              if(f[j].ok&&side(p[i],f[j])>0){
                  dfs(i,j);
                  break;
              }
          }
      }
  }
  void deal(int i,int a,int b){
      int c=to[a][b];
      if(f[c].ok){
          if(side(p[i],f[c])>0)dfs(i,c);
          else{
              add.a=b,add.b=a,add.c=i,add.ok=1;
              to[i][b]=to[a][i]=to[b][a]=n;
              f[n++]=add;
          }
      }
  }
  void dfs(int i,int j){
      f[j].ok=0;
      deal(i,f[j].b,f[j].a);
      deal(i,f[j].c,f[j].b);
      deal(i,f[j].a,f[j].c);
  }
  pt get_myheart(){
      pt s;
      dd v=0;
      for(int i=0;i<n;i++)
          if(f[i].ok){
              dd t=p[f[i].a].V6(p[f[i].b],p[f[i].c]);
              v+=t;
```

```
63                  s=s+(p[f[i].a]+p[f[i].b]+p[f[i].c])*t;
64              }
65          rt s*(1/(v*4));
66      }
67      dd min_dis(){
68          pt t=get_myheart();
69          dd s=1e50;
70          for(int i=0;i<n;i++)
71              if(f[i].ok){
72                  pt tem=p[f[i].a].X(p[f[i].b],p[f[i].c]);
73                  dd ret=fabs(tem*(p[f[i].a]-t)/(tem.abs()));
74                  if(s>ret)s=ret;
75              }
76          rt s;
77      }
78      bool v[N*10];
79      bool coface(int i,int j){
80          rt (p[f[i].a].X(p[f[i].b],p[f[i].c])
81                  ^p[f[j].a].X(p[f[j].b],p[f[j].c])).abs()<eps;
82      }
83      void color(int i){
84          v[i]=0;
85          int a=f[i].a,b=f[i].b,c=f[i].c;
86          if(v[to[b][a]]&&coface(i,to[b][a]))color(to[b][a]);
87          if(v[to[a][c]]&&coface(i,to[a][c]))color(to[a][c]);
88          if(v[to[c][b]]&&coface(i,to[c][b]))color(to[c][b]);
89      }
90      int n_face(){
91          int cnt=0;
92          for(int i=0;i<n;i++)v[i]=f[i].ok;
93          for(int i=0;i<n;i++)
94              if(v[i]){
95                  color(i);
96                  cnt++;
97              }
98          rt cnt;
99      }
100 };
```

2.13 小

2.13.1 点集分割

n*n 枚举所有把点集分成两份等量的方法

```
1  struct pt{
2      int x,y;
3      int id;
4      pt(){}
5      pt(int x,int y):x(x),y(y){}
6      void get(int t){scanf("%d%d",&x,&y);id=t;}
7      pt op+(cpt p)ct{rt pt(x+p.x,y+p.y);}
8      pt op-(cpt p)ct{rt pt(x-p.x,y-p.y);}
9      int op^(cpt p)ct{rt x*p.y-y*p.x;}
10     bool op<(cpt p)ct{rt y<p.y||y==p.y&&x<p.x;}
11 }p[2222];
12 int n,B,W;
13 #define inc(x) (((x)?B:W)++)
14 #define dec(x) (((x)?B:W)--)
15 bool update(int &s,int &e,bool type){
16     pt M=p[e]-p[s],R;
17     int id=-1;
18     bool t;
19     for(int i=0;i<n;i++){
20         if(i==s||i==e)continue;
21         pt U=(p[i]-p[s]);
22         if((M^U)<0){
23             U.x=-U.x,U.y=-U.y;
```

```
24                  if(id==-1||(U^R)>0)R=U,id=i,t=0;
25              }else if(id==-1||(U^R)>0)R=U,id=i,t=1;
26          }
27          if(type)inc(p[e].id);
28          if(t==0){
29              e=s;
30              s=id;
31              if(type)dec(p[s].id);
32          }else{
33              e=id;
34              if(!type)dec(p[e].id);
35          }
36          rt t;
37      }
38      int main(){
39          int T;
40          scanf("%d",&T);
41          while(T--){
42              int n1;
43              scanf("%d",&n1);
44              n=n1<<1;
45              int p1=0,p2=n;
46              for(int i=0;i<n;i++){
47                  p[i].get(i<n1);
48                  if(p[i]<p[p1])p1=i;
49              }
50              p[n]=p[p1]+pt(10,0);
51              update(p1,p2,1);
52              B=0,W=0;
53              while(B+W!=n1-1)
54                  update(p1,p2,1);
55              int s=p1,e=p2,cnt=0,n2=n1>>1;
56              while(true){
57                  if(W==n2&&B+p[s].id==n2)cnt++;
58                  if(W+!p[e].id==n2&&B==n2)cnt++;
59                  if(update(e,s,0))
60                      while(!update(s,e,1));
61                  if(s==p1&&e==p2)break;
62              }
63              printf("%d\n",cnt/2);
64          }
65          rt 0;
66      }
```

2.13.2 二元一次方程

```
1   #include <cstdio>
2   #include <cmath>
3   #include <algorithm>
4   using namespace std;
5   #define cpt ct pt&
6   #define op operator
7   #define dd double
8   #define rt return
9   #define ct const
10  ct dd eps=1e-8;
11  inline int sig(ct dd &x){rt (x>eps)-(x<-eps);}
12  struct pt{
13      dd x,y,r;
14      pt(dd x=0,dd y=0,dd r=0):x(x),y(y),r(r){}
15      void get(){scanf("%lf%lf%lf",&x,&y,&r);}
16      void out(){printf("%lf␣%lf␣%lf\n",x,y,r);}
17      dd getc()ct{rt x*x+y*y-r*r;}
18      bool op==(cpt p)ct{rt x==p.x&&y==p.y&&r==p.r;}
19      bool op<(cpt p)ct{rt r<p.r;}
20      pt op-(cpt p)ct{rt pt(x-p.x,y-p.y);}
```

```
21      dd abs2()ct{rt x*x+y*y;}
22      dd abs()ct{rt sqrt(abs2());}
23  }p[100010];
24  int n;
25  int calc(ct pt& a,ct pt& b,ct pt& c,pt &d){
26      dd a1=2*(a.x-b.x),b1=2*(a.y-b.y),c1=a.getc()-b.getc();
27      dd a2=2*(a.x-c.x),b2=2*(a.y-c.y),c2=a.getc()-c.getc();
28      if(a1*b2==a2*b1){
29          if(a1*c2==a2*c1&&b1*c2==b2*c1)return -2;
30          else return -1;
31      }
32      d.x=(b2*c1-b1*c2)/(b2*a1-b1*a2);
33      d.y=(a2*c1-a1*c2)/(a2*b1-a1*b2);
34      d.r=sqrt((d-a).abs2()-a.r*a.r);
35      if(sig(d.r)==0)return -1;
36      for(int i=0;i<n;i++){
37          if(sig(sqrt((d-p[i]).abs2()-p[i].r*p[i].r)-d.r))return -1;
38      }
39      return 0;
40  }
41  void judge(){
42      if(n==1){
43          puts("-2");
44          return;
45      }
46      if(n==2){
47          if(p[0].x==p[1].x&&p[0].y==p[1].y)
48              puts("-1");
49          else puts("-2");
50          return;
51      }
52      int type;
53      pt d;
54      for(int i=2;i<n;i++){
55          type=calc(p[i-2],p[i-1],p[i],d);
56          if(type==-1){
57              puts("-1");
58              return;
59          }
60          if(type==0){
61              d.out();
62              return;
63          }
64      }
65      puts("-2");
66  }
67  int main() {
68      int T;
69      scanf("%d",&T);
70      while(T--){
71          scanf("%d",&n);
72          for(int i=0;i<n;i++){
73              p[i].get();
74          }
75          sort(p,p+n);
76          n=unique(p,p+n)-p;
77          judge();
78      }
79      return 0;
80  }
```

2.14 老模板，主要是 DP

```
1  //背包变形，有交易限制
2  int f0[100010],f1[100010],cost[60],num[60],c[60][110],w[60][110];
3  int main(){
4      int n,v,tp,i,j,s;
```

```
    while(scanf("%d%d",&n,&v)!=EOF){
        memset(f0,0,sizeof(f0[0])*(v+1));
        for(i=0;i<n;i++){
            scanf("%d%d",&cost[i],&num[i]);
            for(j=0;j<num[i];j++)
                scanf("%d%d",&c[i][j],&w[i][j]);
        }
        for(i=0;i<n;i++){
            memset(f1,-63,sizeof(f1[0])*(v+1));
            for(j=0;j<num[i];j++)
                for(s=v;s>=c[i][j];s--){
                    tp=s-c[i][j];
                    if(f1[s]<f1[tp]+w[i][j])f1[s]=f1[tp]+w[i][j];
                    tp-=cost[i];
                    if(tp>=0&&f1[s]<f0[tp]+w[i][j])f1[s]=f0[tp]+w[i][j];
                }
            for(j=0;j<=v;j++)
                if(f0[j]<f1[j])f0[j]=f1[j];
        }
        int ans=0;
        for(i=0;i<=v;i++)
            if(ans<f0[i])ans=f0[i];
        printf("%d\n",ans);
    }
    return 0;
}
//背包求组合数量
#include<stdio.h>
#include<string.h>
int p[2261],f[4][20010]={0};
int g[20005];
void sieve(){
    int i,j;
    for(i=3;i*i<=20000;i+=2)
        if(g[i]==0){
            for(j=i*i;j<=20000;j+=i)
                g[j]=1;
        }
    j=0;
    for(i=3;i<=20000;i+=2)
        if(g[i]==0)
            p[j++]=i;
}
int main(){
    int n,i,ans,j,x,y;
    sieve();
    f[0][0]=1;
    for(int i=0;i<2261;i++)
        for(int j=3;j>0;j--)
            for(int k=20000;k>=p[i];k--)
                f[j][k]+=f[j-1][k-p[i]];
    while(scanf("%d",&y)!=EOF){
        if(y&1)printf("%d\n",f[3][y]);
        else printf("%d\n",f[2][y]);
    }
    return 0;
}
//排序背包，手里钱钱数不能少于一定值，与转移顺序有关
#include <stdio.h>
#include <string.h>
#include <algorithm>
#define N 5001
using namespace std;
int f[N];
struct bag{
    int p,q,v;
}b[N];
```

```
72 bool operator <(bag x,bag y){
73     return (x.q-x.p)<(y.q-y.p);
74 }
75 int main(){
76     int n,m;
77     while(~scanf("%d%d",&n,&m)){
78         for(int i=0;i<=m;i++)f[i]=0;
79         for(int i=0;i<n;i++)
80             scanf("%d%d%d",&b[i].p,&b[i].q,&b[i].v);
81         sort(b,b+n);
82         for(int i=0;i<n;i++)
83             for(int j=m;j>=b[i].q;j--){
84                 int t=f[j-b[i].p]+b[i].v;
85                 if(f[j]<t)f[j]=t;
86             }
87         printf("%d\n",f[m]);
88     }
89     return 0;
90 }
91 //单调队列
92 #include <stdio.h>
93 #define N 1000010
94 #define max(a,b) (a>b?a:b)
95 int q1[N],q2[N],id1[N],id2[N];
96 int main(){
97     int n,m,k,t;
98     while(~scanf("%d%d%d",&n,&m,&k)){
99         int l1=0,l2=0,r1=-1,r2=-1,ans=0,flag=0;
100        for(int i=0;i<n;i++){
101            scanf("%d",&t);
102            while(l1<=r1&&q1[r1]>t)r1--;
103            q1[++r1]=t,id1[r1]=i;
104            while(l2<=r2&&q2[r2]<t)r2--;
105            q2[++r2]=t,id2[r2]=i;
106            if(q2[l2]-q1[l1]<m||q2[l2]-q1[l1]>k){
107                ans=max(ans,i-flag);
108                if(q2[l2]-q1[l1]>k){
109                    flag++;
110                    if(l1<=r1&&id1[l1]<flag)l1++;
111                    if(l2<=r2&&id2[l2]<flag)l2++;
112                }
113            }
114        }
115        ans=max(ans,n-flag);
116        if(m>k)ans=0;
117        printf("%d\n",ans);
118    }
119    return 0;
120 }
121 //求树中最长路条数
122 #include <stdio.h>
123 #include <vector>
124 using namespace std;
125 #define N 10010
126 int n[N], v[N], tn[N], tv[N], f[N], len[N], ans_n, ans_v;
127
128 struct node {
129     int cost, next;
130
131     node() {
132     }
133
134     node(int x, int y) : cost(x), next(y) {
135     }
136 };
137 vector <node> t[N];
138
```

```
139 void dfs(int x) {
140     f[x] = 1;
141     int s1 = 1, s2 = 1, s3 = 1, m1 = 0, m2 = 0, flag = 0;
142     for (int i = 0; i < t[x].size(); i++) {
143         int j = t[x][i].next;
144         if (!f[j]) {
145             dfs(j);
146             v[j] += t[x][i].cost;
147             if (v[j] > m1)m2 = m1, s2 = s1, m1 = v[j], s1 = n[j], s3 = s1 *
                    s1, flag = 1;
148             else if (v[j] == m1)s1 += n[j], s3 += n[j] * n[j], flag = 0;
149             else if (v[j] > m2)m2 = v[j], s2 = n[j];
150             else if (v[j] == m2)s2 += n[j];
151         }
152     }
153     if (flag) tn[x] = s1 * s2, tv[x] = m1 + m2;
154     else tn[x] = (s1 * s1 - s3) / 2, tv[x] = m1 + m1;
155     n[x] = s1, v[x] = m1;
156     if (tv[x] > ans_v)ans_v = tv[x], ans_n = tn[x];
157     else if (tv[x] == ans_v)ans_n += tn[x];
158 }
159
160 int main() {
161     int m, x, y, a;
162     while (~scanf("%d", &m)) {
163         for (int i = 1; i <= m; i++)t[i].clear(), f[i] = 0;
164         for (int i = 1; i < m; i++) {
165             scanf("%d%d%d", &x, &y, &a);
166             t[x].push_back(node(a, y));
167             t[y].push_back(node(a, x));
168         }
169         ans_v = ans_n = 0;
170         dfs(1);
171         printf("%d␣%d\n", ans_v, ans_n);
172     }
173     return 0;
174 }
175 //各种背包
176 #include <stdio.h>
177 #include <string.h>
178 #define max(a,b) ((a)>(b)?(a):(b))
179 int n,v,m,t,w,c,f[101][101];
180 int main(){
181     while(~scanf("%d%d",&n,&v)){
182         memset(f,-31,sizeof(f));
183         f[0][0]=0;
184         for(int i=1;i<=n;i++){
185             scanf("%d%d",&m,&t);
186             while(m--){
187                 scanf("%d%d",&w,&c);
188                 if(t)for(int j=0;j<=v;j++)
189                     f[i][j]=max(f[i][j],f[i-1][j]);
190                 for(int j=v;j>=w;j--){
191                     if(t==0)f[i][j]=max(f[i][j],max(f[i-1][j-w],f[i][j-w])+c)
                            ;
192                     else if(t==1)f[i][j]=max(f[i][j],f[i-1][j-w]+c);
193                     else f[i][j]=max(f[i][j],f[i][j-w]+c);
194                 }
195             }
196         }
197         int ans=-1;
198         for(int i=0;i<=v;i++)ans=max(ans,f[n][i]);
199         printf("%d\n",ans);
200     }
201     return 0;
202 }
203 //以差值做状态
```

```
#include <stdio.h>
#include <string.h>
int T,cs=1,n,c,f[101][2001];
inline int max(int a,int b,int c){
    a=(a>c)?a:c;
    return a>b?a:b;
}
int main(){
    scanf("%d",&T);
    while(T--){
        scanf("%d",&n);
        memset(f,-31,sizeof(f));
        f[0][0]=0;
        int s=0,flag=0;
        for(int i=1;i<=n;i++){
            scanf("%d",&c);
            s+=c;
            if(!c)flag=1;
            for(int j=0;j<=s;j++)
                if(c<j)
                    f[i][j]=max(f[i-1][j],f[i-1][j+c]+c,f[i-1][j-c]);
                else
                    f[i][j]=max(f[i-1][j],f[i-1][c-j]+c-j,f[i-1][j+c]+c);
        }
        if(!f[n][0]&&!flag)f[n][0]=-1;
        printf("Case␣%d:␣%d\n",cs++,f[n][0]);
    }
    return 0;
}
//四边不等等式
#include<iostream>
using namespace std;
const int INF=1000000000;
const int maxn=1005;
int dp[maxn][maxn],s[maxn][maxn],a[maxn],n,x[maxn],y[maxn];
typedef struct point
{
    int x,y;
    point(){};
    point(int xx,int yy){x=xx;y=yy;}
    }point;
inline int cost(point a,point b){return abs(a.x-b.x)+abs(a.y-b.y);}
int main()
{
    int i,j,k,t,w;
    while(scanf("%d",&n)!=EOF)
    {
    for(i=1;i<=n;i++)scanf("%d%d",&x[i],&y[i]);
    for(i=1;i<=n;i++){dp[i][i]=0;s[i][i]=i;}

    for(t=2;t<=n;t++)
    {
        for(i=1;i<=n-t+1;i++)
        {
            j=i+t-1;dp[i][j]=INF;
            for(k=s[i][j-1];k<=s[i+1][j];k++)
            {
                w=cost(point(x[i],y[k-1]),point(x[k],y[j]));
                if(dp[i][j]>dp[i][k-1]+dp[k][j]+w)
                {
                   dp[i][j]=dp[i][k-1]+dp[k][j]+w;
                   s[i][j]=k;
                   }
                }
            }
        }
    printf("%d\n",dp[1][n]);
```

```
271             }
272         return 0;
273 }
274 //点24
275 char s[4];
276 const int n=1<<4;
277 set<double>f[n];
278 set<double>::iterator ii,jj,it;
279 int main(){
280     while(~scanf("%s",s)){
281         for(int i=0;i<n;i++)f[i].clear();
282         for(int i=0,t;i<4;i++){
283             if(s[0]<='9'&&s[0]>'1')t=s[0]-'0';
284             else if(s[0]=='1')t=10;
285             else if(s[0]=='A')t=1;
286             else if(s[0]=='J')t=11;
287             else if(s[0]=='Q')t=12;
288             else t=13;
289             f[1<<i].insert(t);
290             if(i<3)scanf("%s",s);
291         }
292         for(int i=1;i<n;i++)
293             for(int j=1;j<=(i>>1);j++)
294                 if((i&j)==j){
295                     for(ii=f[j].begin();ii!=f[j].end();ii++)
296                         for(jj=f[i^j].begin();jj!=f[i^j].end();jj++){
297                             f[i].insert(*ii+*jj);
298                             f[i].insert(*ii-*jj);
299                             f[i].insert(*jj-*ii);
300                             f[i].insert((*ii)*(*jj));
301                             if(*ii)f[i].insert(*jj/(*ii));
302                             if(*jj)f[i].insert(*ii/(*jj));
303                             if(i==n-1){
304                                 it=f[n-1].find(24);
305                                 if(it!=f[n-1].end()){
306                                     puts("YES");
307                                     goto loop;
308                                 }
309                             }
310                         }
311                 }
312         puts("NO");
313 loop:   ;
314     }
315 }
316 //树形多重背包，合并思想，，。updownthe more the better
317 int v[201],f[201][201];
318 vector<int > t[201];
319 inline int max(int a,int b){
320     return a>b?a:b;
321 }
322 void dfs(int i,int c){
323     if(c<=0)return;
324     for(int r=0;r<t[i].size();r++){
325         int j=t[i][r];
326         for(int k=0;k<c;k++)f[j][k]=f[i][k];
327         dfs(j,c-1);
328         for(int k=1;k<=c;k++)f[i][k]=max(f[i][k],f[j][k-1]+v[j]);
329     }
330 }
331 int main(){
332     int n,m,x;
333     while(scanf("%d%d",&n,&m),n||m){
334         for(int i=0;i<=n;i++){
335             t[i].clear();
336             for(int j=0;j<=m;j++)f[i][j]=0;
337         }
```

```
338         for(int i=1;i<=n;i++){
339             scanf("%d%d",&x,&v[i]);
340             t[x].push_back(i);
341         }
342         dfs(0,m);
343         printf("%d\n",f[0][m]);
344     }
345     return 0;
346 }
347 //对叶子进行优化，省去叶子节点的空间
348 #include <stdio.h>
349 int c[100001],v[100001],f[501][10001],t[501],p[101000][2],q;
350 #define max(a,b) (a>b?a:b)
351 #define push(a,b) p[q][0]=b;p[q][1]=t[a];t[a]=q++
352 void dfs(int i,int s){
353     if(s<1)return;
354     int q=t[i];
355     while(q){
356         int j=p[q][0];
357         if(j<501&&t[j]){
358             for(int k=s-c[j];k>=0;k--)f[j][k]=f[i][k];
359             dfs(j,s-c[j]);
360             for(int k=c[j];k<=s;k++)f[i][k]=max(f[i][k],f[j][k-c[j]]+v[j]);
361         }else for(int k=s;k>=c[j];k--)f[i][k]=max(f[i][k],f[i][k-c[j]]+v[j]);
362         q=p[q][1];
363     }
364 }
365 int main(){
366     int n,g,x;
367     while(~scanf("%d%d",&n,&g)){
368         for(int i=500;i>=0;i--){
369             for(int j=0;j<=g;j++)f[i][j]=0;
370             t[i]=0;
371         }
372         q=1;
373         for(int i=1;i<=n;i++){
374             scanf("%d%d%d",&c[i],&v[i],&x);
375             if(x==i)x=0;
376             push(x,i);
377         }
378         dfs(0,g);
379         printf("%d\n",f[0][g]);
380 
381     }
382     return 0;
383 }
384 //状态压缩，彩色石头
385 int dp[101][6][160];
386 int max(int a,int b){return a>b?a:b;}
387 int main(){
388     int m,k;
389     while(scanf("%d%d",&m,&k),m||k){
390         memset(dp,0,sizeof(dp));
391         int n=1<<k,ans=0,a;
392         for(int i=1;i<=m;i++){
393             scanf("%d",&a);a--;
394             for(int j=0;j<k;j++)
395                 for(int s=0;s<n;s++)
396                     if((s>>a)&1){
397                         if(a==j) dp[i][j][s]=max(dp[i][j][s],dp[i-1][j][s]+1)
                                ;
398                         else dp[i][j][s]=max(dp[i][j][s],dp[i-1][j][s]);
399                     }
400                     else{
401                         dp[i][a][s|(1<<a)]=max(dp[i][a][s|(1<<a)],dp[i-1][j][
                            s]+1);
402                         dp[i][j][s]=max(dp[i][j][s],dp[i-1][j][s]);
```

```
                    }
            }
            for(int i=0;i<k;i++)
                for(int j=0;j<n;j++)
                    ans=max(ans,dp[m][i][j]);
            printf("%d\n",m-ans);
        }
        return 0;
}
//状态压缩，瓷砖覆盖dp
const int N=1<<11;
__int64 y[N],x[N];
int n,m,t;
bool check(int x){
        while(x){
            if(x&1){
                x>>=1;
                if(x&1)x>>=1;
                else return 0;
            }else x>>=1;
        }
        return 1;
}
bool judge(int now,int last){
        int tmp=t&(~last);
        if((now&tmp)==tmp&&check(now-tmp))return 1;
        return 0;
}
int main(){
        while(scanf("%d%d",&n,&m),n||m){
            if((n&1)&&(m&1)){puts("0");continue;}
            if(n<m)n^=m,m^=n,n^=m;
            int s=1<<m;
            t=s-1;
            __int64 sum=0;
            for(int i=0;i<s;i++)y[i]=check(i);
            for(int i=1;i<n;i++){
                for(int j=0;j<s;j++)
                    for(int k=0;k<s;k++)
                        if(judge(j,k))x[j]+=y[k];
                memcpy(y,x,sizeof(x[0])*s);
                memset(x,0,sizeof(x[0])*s);
            }
            for(int i=0;i<s;i++)
                if(judge(0,i))sum+=y[i];
            printf("%I64d\n",sum);
        }
        return 0;
}
//一个整数分解为 ，。。。这种12,4, 2 的倍数的分解方式有多少种。
int f[1000001]={0},t;
int main(){
            f[0]=1;
            scanf("%d",&t);
            for(int i=1;i<=t;i++){
                    if(i&1)f[i]=f[i-1];
                    else f[i]=f[i-2]+f[i>>1];
                    if(f[i]>=1000000000)f[i]%=1000000000;
            }
            printf("%d\n",f[t]);
            return 0;
}
//最长公共子序列
int m,n,r[300],s[300];
char a[300],b[300];
int main(){
        while(scanf("%s%s",a,b)!=EOF){
```

```
        m=strlen(a);n=strlen(b);
        memset(r,0,sizeof(r));s[0]=0;
        for(int i=1;i<=m;i++){
            for(int j=1;j<=n;j++)
                if(a[i-1]==b[j-1]) s[j]=r[j-1]+1;
                    else s[j]=r[j]>s[j-1]?r[j]:s[j-1];
            for(int k=1;k<=n;k++)r[k]=s[k];
        }
        printf("%d\n",r[n]);
    }
    return 0;
}
#include <stdio.h>
#include <iostream>
#include <algorithm>
using namespace std;
typedef long long LL;
const LL oo=1ll<<60;
struct point{
    int x,y;
    void get(){scanf("%d%d",&x,&y);}
    bool operator<(const point &p)const{
        return x>p.x||x==p.x&&y>p.y;
    }
}a[50001],b[50001];
int s[101][50001];
LL f[101][50001];
//f[i][j]=min(f[i-1][k]+w(k+1,j))
//s[i-1][j]<=s[i][j]<=s[i][j+1]
int main(){
    int cnt,m,k;
    while(~scanf("%d%d",&cnt,&m)){
        for(int i=0;i<cnt;i++)b[i].get();
        sort(b,b+cnt);
        int n=0;
        a[++n]=b[0];
        for(int i=1;i<cnt;i++)
            if(b[i].y>a[n].y)
                a[++n]=b[i];
        for(int i=1;i<=m;i++)s[i][n+1]=n-1;
        for(int i=1;i<=n;i++){
            f[1][i]=(LL)a[1].x*a[i].y;
            s[1][i]=1;
        }
        for(int i=2;i<=m;i++)
            for(int j=n;j>=i;j--){
                f[i][j]=+oo;
                int l=max(s[i-1][j],i-1),r=min(s[i][j+1],j-1);
                for(int k=l;k<=r;k++){
                    LL t=f[i-1][k]+(LL)a[k+1].x*a[j].y;
                    if(f[i][j]>t){
                        f[i][j]=t;
                        s[i][j]=k;
                    }
                }
            }
        LL ans=+oo;
        for(int i=m;i>0;i--)
            ans=min(ans,f[i][n]);
        cout<<ans<<endl;
    }
    return 0;
}
```