## Contents

1	数据		1
	1.1	点分治	1
	1.2	AC自动机	2
	1.3	后缀数组	3
<b>2</b>	数学		3
	2.1	线性基	3
	2.2	类欧几里得	5
	2.3	高斯消元法实数方程	8
	2.4	高斯消元法模方程	8
	2.5	扩展欧几里得	9
	2.6	FWT异或,与,或	10
3	图论		11
•	–	tarjan	11
	• • •	3.1.1 有向图强连通分量	11
		3.1.2 点双联通分量	12
		3.1.3 边双联通分量	13
	3.2	网络流Dinic	14
	3.3	费用流	15
4	小算	法	16
-	4.1	跨平台大随机数	16
	4.2	读入优化	16
	4.3	KMP算法	16
	4.4	扩展KMP	17
_	` I. <i>65</i> 7	. n. <i>l</i> ar	17
5	计算 5.1	儿們 凸包	$\frac{17}{17}$
	$5.1 \\ 5.2$	定义	18
	$\frac{5.2}{5.3}$	半平面交	$\frac{10}{22}$
	5.4	圆与多边形交集	23
	$5.4 \\ 5.5$	三角形面积并	$\frac{23}{23}$
	5.6	<u> </u>	$\frac{25}{26}$
6		<b>计算几何</b>	29
	6.1	基本定义	29
	6.2	一些补充	29
7	bitse	et用法	30

# 1 数据结构

## 1.1 点分治

```
void Size(int x, int fa)

to void Size(int x, int fa)

for pt, next;

size[x]=1;

for (pt=first[x]; pt; pt=e[pt]. next)

for next=e[pt].to;
```

```
(next=fa | | vis [next]) continue;
              Size (next,x);
9
              size[x] += size[next];
11
12
   int Center(int x, int fa, int ori)
13
14
   \mathbf{int} \ \mathrm{pt} \ , \mathrm{next} \ ;
15
         for (pt=first [x]; pt; pt=e[pt].next)
16
17
              next=e[pt].to;
18
              if (next=fa || vis [next] || 2 * size [next] <= size [ori]) continue;
19
              return Center (next, x, ori);
20
21
        return x;
22
23
   void DFS(int x)
24
25
   int pt, next, i, j, low, high;
26
        Size(x,x);
27
        x = Center(x, x, x);
28
         vis[x] = true;
29
        for (pt=first [x]; pt; pt=e[pt].next)
30
31
              next=e[pt].to;
32
              if (vis[next]) continue;
33
             DFS(next);
34
35
         vis[x] = false;
36
37
```

### 1.2 AC自动机

```
void Build_AC()
2
          low, high, i;
   int
   Node *temp:
          low = 0;
          high=-1;
6
          for (i=0; i<26; i++)
               (root \rightarrow son[i]) que[++high] = root \rightarrow son[i];
          for (; low<=high; low++)
               for (i=0; i<26; i++)
10
               i f
                     (que [low]->son [i])
11
12
                     que[++high]=que[low]->son[i];
13
                     for (temp=que [low]; temp!=root; temp=temp->fail)
14
                          (\text{temp->} \text{fail ->} \text{son} [i])
15
16
                          que[high] -> fail = temp -> fail -> son[i];
17
                          break;
18
                     }
19
```

```
20 }
21 }
```

### 1.3 后缀数组

```
bool Same(int *st, int a, int b, int len)
   {
2
         return st [a]==st[b]\&\&st[a+len]==st[b+len];
3
   void DA(int m=1000)
6
   int
         cnt, i, len, *x=arr1, *y=arr2;
7
         memset(arr1,127,sizeof(arr1));
         memset(arr2, 127, sizeof(arr2));
9
         for (i=0; i \le m; i++) sum[i]=0;
10
         for (i=1;i<=n;i++) sum[x[i]=st[i]]++;
11
         for (i=1; i \le m; i++) sum[i]+=sum[i-1];
12
         for (i=n; i>=1; i--) sa [sum[x[i]]--]=i;
13
         for (cnt=0, len=1; cnt < n; len < <=1, m=cnt)
15
              for (cnt=0, i=n-len+1; i \le n; i++) y[++cnt]=i;
16
              for (i=1; i \le n; i++)
17
              i f
                   (sa[i]>len) y[++cnt]=sa[i]-len;
18
              for (i=0; i \le m; i++) sum[i]=0;
19
              for (i=1;i \le n;i++) sum[x[y[i]]]++;
              for (i=1; i \le m; i++) sum [i]+=sum [i-1];
21
              for (i=n; i>=1;i--) sa [sum[x[y[i]]]--]=y[i];
22
              swap(x,y);
23
              for (cnt=1,x[sa[1]]=1,i=2;i \le n;i++)
24
                  (Same(y, sa[i-1], sa[i], len)) \times [sa[i]] = cnt; else \times [sa[i]]
25
                  || = + + cnt |
26
         for (i=1; i \le n; i++) rank [sa[i]] = i;
27
   }
28
   void Height()
29
30
   int
31
         for (len=0, i=1; i \le n; height [rank[i++]]=len)
32
33
              i f
                   (len) len --;
34
                   (rank[i]==1) continue;
35
              for (; st [i+len] = st [sa [rank[i]-1]+len]; len++);
37
38
```

# 2 数学

### 2.1 线性基

```
<sup>1</sup> #include < bits / stdc++.h>
```

```
using namespace std;
   #define B 30
   #define N 10050
   const int allset=(1 << B) -1;
   int a[N];
   struct LB
10
11
           int mat[B], cnt;
12
           multiset <int> st;
13
           LB(){}
14
           void clear()
15
16
                  st.clear();
                 cnt=0;
18
                 memset(mat,0, sizeof(mat));
19
20
           void add(int x)
21
22
           int
                 i , j ;
23
                  for (i=B-1; i>=0; i--)
24
                  i f
                      ((x>>i)&1)
25
26
                      i f
                           (mat[i]) x^=mat[i]; else
27
28
                           cnt++;
29
                           mat[i]=x;
30
                           break;
31
32
33
           void fix()
35
           int
                 i , j ;
37
                  for (i=0; i< B; i++)
                  i f
                      (mat[i])
39
40
                      for (j=i+1; j < B; j++)
41
                      if ((mat[j]>>i)&1) mat[j]^=mat[i];
42
43
44
           void preset () //正确性待定
45
46
           int
                 i ;
47
                  fix();
48
                  for (i=0; i< B; i++)
49
                      (mat[i]) st.insert(mat[i]);
50
                 kth(int k)//正确性待定
           int
52
53
           int
                i, ans;
54
            multiset < int > :: iterator it;
```

```
(k <= 0 | |k > (1 << cnt) -1) return 0; // \Xi M
                  for (ans=i=0, it=st.begin(); it!=st.end(); it++,i++)
57
                      ((k>>i)\&1) ans^=(*it);
                  return ans;
59
60
            int
                 getmax()
61
62
            int
                 i, ans;
63
                  fix();
64
                  ans=0;
65
                  for (i=B-1; i>=0; i--)
66
                     (ans^mat[i]>ans) ans^mat[i];
67
                  return ans;
68
69
            tree [N*10];
70
```

### 2.2 类欧几里得

$$\textstyle\sum_{x=0}^n x^{k1}(\left\lfloor\frac{ax+b}{c}\right\rfloor)^{k2}$$

```
1 #include < bits / stdc++.h>
  using namespace std;
  #define pb push_back
   typedef long long 11;
   const int MOD=1e9+7;
   inline ll qp(ll a, ll b)
        11 x=1; a\% = MOD;
        while(b)
9
10
             \mathbf{i} \mathbf{f} (b \& 1) = x * a MOD;
11
             a=a*a/MOD; b>>=1;
12
13
        return x;
14
   }
15
   namespace Lagrange {
16
   11 x[23333], y[23333], a[23333], g[23333], h[23333], p[23333]; int N;
17
   void work()
18
   {
19
        for (int i = 0; i < N; ++i) a[i] = 0;
20
        g[0] = 1;
21
        for (int i = 0; i < N; ++i)
22
23
             for(int _=0;_<=i;++_)
24
                  h[-+1]=g[-]; h[0]=0;
25
             for(int _=0;_<=i;++_)
26
                  h[_{-}] = (h[_{-}] - g[_{-}] * (11)x[i]) MOD;
27
             for (int _=0; _<=i+1;++_) g[_]=h[_];
28
29
        for (int i=0; i< N; ++i)
30
31
             for (int j=0; j<=N;++j) p[j]=g[j];
32
             for (int j=N; j;--j)
33
```

```
p[j-1]=(p[j-1]+p[j]*(11)x[i])%MOD;
34
              11 s=1;
35
              for(int j=0; j<N;++j) if(i!=j)
                   s=s*(x[i]-x[j])MOD;
37
              s=y [ i ] * qp ( s ,MOD–2)%MOD;
38
              for(int _=0;_<N;++_)
39
                  a = -1 = (a = -1) + p = -1 \times s )%MOD;
40
41
42
   vector<int> feed (vector<int> v)
43
44
        N=v. size();
45
        for(int i=0;i<N;++i) x[i]=i,y[i]=v[i];
46
        work(); v.clear();
47
        for (int i=0; i<N;++i) v.pb(a[i]);
48
        while(v.size()&&!v.back()) v.pop_back();
49
        return v;
50
51
       calc (vector < int > &v, ll xx)
   11
52
53
         11 s=0, gg=1; xx\%=MOD;
54
        for (int i=0; i<N;++i)
55
              s = (s+gg*v[i])MOD, gg=gg*xxMOD;
56
        return s;
57
58
   }
59
   using Lagrange::feed;
60
   using Lagrange::calc;
61
   //ps[k]=\sum_{\{i=0\}}\ximus i\ximus k
   vector < int > ps[2333];
63
   //rs \lceil k \rangle = \langle sum_{-} \{ i = 0 \} \hat{x} ((i+1) \hat{k} - i \hat{k}) \rangle
64
   vector < int > rs[2333];
65
   struct arr{ll p[11][11];};
   11 C[233][233];
67
   arr calc(ll a, ll b, ll c, ll n)
69
70
         arr w;
         if(n==0) a=0:
71
         if(a==0||a*n+b<c)
72
73
              for(int i=0; i <=10;++i)
74
75
                   11 t=calc(ps[i],n),s=b/c;
76
                   for (int j=0; i+j \le 10; ++j)
77
                       w.p[i][j]=t, t=t*sMOD;
78
79
             return w;
80
81
        for (int i=0; i <=10;++i)
82
             w.p[i][0] = calc(ps[i], n);
        if(a>=c||b>=c)
84
85
              arr t = calc(a\%c,b\%c,c,n);
86
              11 p=a/c, q=b/c;
```

```
for (int i=0; i <=10;++i)
                      for (int j=1; i+j \le 10; ++j)
 89
                            11 s=0, px=1;
 91
                           for (int x=0;x<=j;++x,px=px*p%MOD)
 92
 93
                                 11 qy=1;
94
                                 for (int y=0;x+y<=j;++y,qy=qy*q%MOD)
 95
 96
                                       //x \hat{(i)} (px) \hat{x} q \hat{y} ?? \hat{(j-x-y)}
97
                                       s + = px * qy MOD*C[j][x] MOD*C[j-x][y]
98
                                      MOD*t.p[i+x][j-x-y]; sM=MOD;
99
100
101
                           w.p[i][j]=s;
102
103
                return w;
104
105
          ll m=(a*n+b)/c;
106
          \operatorname{arr} \operatorname{t=calc}(c, c-b-1, a, m-1);
107
          for (int i=0; i <=10;++i)
108
                for (int j=1; i+j \le 10; ++j)
109
110
                      11 \operatorname{s=calc}(\operatorname{rs}[j], m-1) * \operatorname{calc}(\operatorname{ps}[i], n) \% MOD;
111
                      for (int p=0; p< j; ++p)
112
113
                           for (unsigned q=0; q < ps[i]. size ();++q)
114
115
                                 ll v=C[j][p]*ps[i][q]%MOD;
116
                                 //v*t^p*((tc+c-b-1)/a)^q
117
                                 s=t.p[p][q]*v; s=MOD;
118
119
120
                     w.p[i][j] = s MOD;
121
122
          return w;
123
124
    int T, n, a, b, c, k1, k2;
125
    int main()
126
     {
127
           freopen ("a.in", "r", stdin);
128
          freopen ("a.out", "w", stdout);
129
          for (int i=0; i <=230;++i)
130
131
                C[i][0] = 1;
132
                for (int j=1; j <= i; ++j)
133
                     C[i][j] = (C[i-1][j-1] + C[i-1][j]) \text{MOD};
134
135
          for (int i=0; i <=10;++i)
136
137
                11 \text{ sp}=0, \text{sr}=0; \text{ vector} < \text{int} > \text{p,r};
138
                for (int j=0; j <=20;++j)
139
                      sp + = qp(j, i), sr + = qp(j+1, i) - qp(j, i),
140
                      sp\%=MOD, sr\%=MOD, p.pb(sp), r.pb(sr);
141
```

```
ps[i] = feed(p); rs[i] = feed(r);
142
143
         scanf("%d",&T);
144
         \mathbf{while}(\mathbf{T}--)
145
146
               scanf("%d%d%d%d%d%d%d"),
147
              &n,&a,&b,&c,&k1,&k2);
148
               arr s=calc(a,b,c,n);
149
               int p=s.p[k1][k2];
150
               p = (p MOD + MOD) MOD;
151
               printf("%d\n",p);
152
         }
153
154
```

## 2.3 高斯消元法实数方程

```
void Gauss(int n,int m)
   {
2
   int
        i , j , k , t ;
3
   double mul;
         for (i=j=1; i \le 1; i \le m; i++, j++)
6
              for (k=i+1;k \le n;k++)
                  (abs(mat[k][j])>abs(mat[i][j]))
                  for (t=1;t<=m+1;t++) swap(mat[i][t],mat[k][t]);
             i f
                  (abs(mat[i][j])<eps)
10
             {
11
                  i --:
12
                  continue;
13
             for (k=i+1;k \le n;k++)
15
16
                  mul=mat[k][j]/mat[i][j];
17
                  for (t=1;t=m+1;t++) \max[k][t]=\max[i][t]*mul;
18
19
20
         for (i=n; i>=1;i−−)//表示那个变量是否确定 solved
21
22
              for (j=1; j \le m; j++)
23
                  (abs(mat[i][j])>eps) break;
              i f
                  (j>m) continue;
25
              solved [j]=true;
26
             ans[j]=mat[i][m+1];
27
              for (k=j+1;k \le m;k++)
28
                  (abs(mat[i][k])>eps\&\&!solved[k]) solved[j]=false;
29
              for (k=j+1; k \le m; k++) ans [j] = ans [k] * mat [i] [k];
30
              ans[j]/=mat[i][j];
31
32
33
```

### 2.4 高斯消元法模方程

```
long long Pow(long long x,long long y)
2
         if (y==0) return 1;
   long long t=Pow(x, y/2);
              (y\&1) return t*t\mbox{mod}*x\mbox{mod};
         return t * t \mod;
   }
   void Gauss (long long n, long long m)
9
   long long i, j, k, t, lcm, muli, mulk;
10
         for (i=j=1; i \le m \le j \le m; i++, j++)
11
12
              for (k=i; k<=n; k++)
13
                   (mat[k][j])
              i f
15
                   for (t=1;t<=m+1;t++) swap(mat[k][t],mat[i][t]);
16
                   break:
17
18
              i f
                   (mat[i][j]==0)
19
              {
20
                   i --:
21
                   continue;
22
23
              for
                  (k=i+1;k\leq n;k++)
24
              i f
                   (mat[k][j])
25
26
                   lcm=mat[k][j]*mat[i][j]/Gcd(mat[k][j],mat[i][j]);
27
                   muli=lcm/mat[i][j];
28
                   mulk=lcm/mat[k][j];
29
                   for (t=1;t=m+1;t++)
30
31
                       mat[k][t]=mat[k][t]*mulk-mat[i][t]*muli;
32
                       mat[k][t] = (mat[k][t]\%mod+mod)\%mod;
33
34
              }
35
36
         for (i=n; i>=1; i--)
37
38
              for (j=1; j \le m; j++)
39
                   (mat[i][j]) break;
40
                   (j>m) continue;
41
              ans[j]=mat[i][m+1];
42
              for (k=j+1; k \le m; k++) ans [j] = ans [k] * mat [i] [k];
43
              ans[j] = (ans[j] * Pow(mat[i][j], mod-2)%mod+mod)%mod;
         }
45
46
```

### 2.5 扩展欧几里得

求一组整数解且防爆

```
void GCD.EX(long long A, long long &x, long long B, long long &y, long long C)
```

```
long long xx, yy, temp, gcd;
3
            (B = 0)
         i f
5
             x = C;
6
             y = 0;
             else
             GCD_EX(B, xx, A \% B, yy, C);
10
             x = yy;
11
             temp = 1 - x / C * A;
12
             gcd = GCD(temp, B);
13
             temp /= gcd;
14
             B /= gcd;
15
             y = C / B * temp;
16
17
         return;
18
19
```

### 2.6 FWT异或,与,或

```
void fwtXor(int* a, int len) {
       if(len = 1) return;
2
       int h = len \gg 1;
3
       fwtXor(a, h);
       fwtXor(a + h, h);
       for(int i = 0; i < h; ++i) {
6
           int x1 = a[i];
           int x2 = a[i + h];
           a[i] = (x1 + x2) \% mod;
           a[i + h] = (x1 - x2 + mod) \% mod;
10
11
   }
12
   void ifwtXor(int* a, int len) {
13
       if(len = 1) return;
14
       int h = len \gg 1;
15
       for(int i = 0; i < h; ++i)
16
           int y1 = a[i];
17
           int y2 = a[i + h];
18
           a[i] = 111*(y1 + y2) * div2 \% mod;
19
           a[i + h] = 111*(y1 - y2 + mod) * div2 \% mod;
20
21
       ifwtXor(a, h);
22
       ifwtXor(a + h, h);
23
24
   void fwtAnd(int* a, int len) {
25
       if(len = 1) return;
26
       int h = len \gg 1;
27
       fwtAnd(a, h);
28
       fwtAnd(a + h, h);
29
       for(int i = 0; i < h; ++i) {
30
           int x1 = a[i];
31
```

```
int x2 = a[i + h];
            a[i] = (x1 + x2) \% \text{ mod};
33
            a[i + h] = x2;
       }
35
36
   void ifwtAnd(int* a, int len) {
37
        if(len = 1) return;
38
       int h = len \gg 1;
39
       for(int i = 0; i < h; ++i)
40
            int y1 = a[i];
41
            int y2 = a[i + h];
42
            a[i] = (y1 - y2 + mod) \% mod;
43
            a[i + h] = y2;
44
45
       ifwtAnd(a, h);
46
       ifwtAnd(a + h, h);
47
48
   void fwtOr(int* a, int len) {
49
       if(len = 1) return;
50
       int h = len \gg 1;
51
       fwtOr(a, h);
52
       fwtOr(a + h, h);
53
       for(int i = 0; i < h; ++i) {
54
            int x1 = a[i];
55
            int x2 = a[i + h];
56
            a[i] = x1;
57
            a[i + h] = (x1 + x2) \% mod;
58
59
   }
60
   void ifwtOr(int* a, int len) {
61
       if(len = 1) return;
62
       int h = len \gg 1;
63
       for(int i = 0; i < h; ++i) {
            int y1 = a[i];
65
            int y2 = a[i + h];
            a[i] = y1;
67
            a[i + h] = (y2 - y1 + mod) \% mod;
69
       ifwtOr(a, h);
70
       ifwtOr(a + h, h);
71
72
```

# 3 图论

#### 3.1 tarjan

#### 3.1.1 有向图强连通分量

```
void DFS(int x)

{
    int     pt, next;
    dfn[x]=low[x]=++times;
    sk[++tp]=x;
```

```
instack[x]=true;
            for (pt=first [x]; pt; pt=e[pt].next)
                      next=e[pt].to;
                      if (!dfn[next])
10
            {
11
                 DFS(next);
12
                 low[x]=min(low[x], low[next]);
13
                 else
14
                 (instack[next]) low[x]=min(low[x], dfn[next]);
            i f
15
16
            i f
                 (low[x]==dfn[x])
17
18
                      tot++;
19
                     for (; tp;)
20
21
                          instack[sk[tp]] = false;
22
                               belong [sk[tp--]]=tot;
23
                                   (sk[tp+1]==x) break;
24
                     }
26
27
   for (i=1;i<=n;i++)
28
        (!dfn[i]) DFS(i);
   i f
29
```

#### 3.1.2 点双联通分量

```
void DFS(int x,int fa)
2
   int
         pt, next;
3
         vis[x] = true;
         dfn[x] = low[x] = ++times;
5
         sk[++tp]=x;
         for (pt=first [x]; pt; pt=e[pt].next)
              next=e[pt].to;
                  (e[pt].id=fa) continue;
10
              i f
                  (! vis [next])
11
12
                  DFS(next, e[pt].id);
13
                  low[x]=min(low[x], low[next]);
14
                       (low[next]>=dfn[x])
                  i f
15
                  {
16
                       tot++;
17
                       vec[tot].clear();
18
                       for (; tp;)
19
20
                           vec [tot].push_back(sk[tp]);
21
22
                           if (sk[tp+1]==next) break;
23
24
                       vec[tot].push_back(x);
25
                  }
26
```

```
else
27
                        (dfn[next]>last) low[x]=min(low[x],dfn[next]);
                  i f
28
29
30
    for (i=1; i \le n; i++)
31
    i f
          (! vis [i])
32
33
        DFS(i,0);
34
        last=times;
35
         i f
             (tp)
36
37
         {
               tot++;
38
               vec[tot].clear();
39
              \mbox{ for } \ (\,i\,{=}1; i{<}{=}tp\,; \,i\,{+}{+}) \ vec\,[\,tot\,]\,.\,push\_back\,(\,sk\,[\,i\,]\,)\;;
40
41
42
43
```

#### 3.1.3 边双联通分量

```
void DFS(int x,int fa)
2
   int
         pt, next;
3
         vis[x] = true;
         dfn[x] = low[x] = ++times;
         sk[++tp]=x;
         for (pt=first [x]; pt; pt=e[pt].next)
         {
              next=e[pt].to;
                   (e[pt].id=fa) continue;
10
              i f
                   (! vis [next])
11
              {
12
                  DFS(next, e[pt].id);
13
                  low[x]=min(low[x], low[next]);
14
                       (low[next]>dfn[x])
                   i f
15
                   {
16
                        tot++;
17
                       vec[tot].clear();
18
                       for (; tp;)
19
20
                            vec [tot].push_back(sk[tp]);
21
22
                            if (sk[tp+1]==next) break;
23
                       }
24
25
                   else
26
              i f
                   (dfn[next]>last) low[x]=min(low[x], dfn[next]);
27
         }
29
   for (i=1; i \le n; i++)
30
   i f
        (! vis [i])
31
32
       DFS(i,0);
33
```

```
last=times;
34
       i f
           (tp)
35
36
            tot++;
37
            vec[tot].clear();
38
            for (i=1;i<=tp;i++) vec[tot].push_back(sk[i]);
39
            tp=0;
40
41
   }
42
```

### 3.2 网络流Dinic

```
struct Edge
1
   {
2
           int to,flow,next;
           e[1000050];
   void Add(int x,int y,int z)
   {
6
         e[++now]. to=y;
         e [now]. flow=z;
         e[now].next=first[x];
         first [x]=now;
10
11
   bool Level()
12
   {
13
         low, high, pt, next;
   int
14
         memset (level, -1, sizeof(level));
15
         q[low=high=0]=S;
16
         level[S]=0;
17
         for (; low<=high; low++)
18
             for (pt=first [q[low]]; pt!=-1; pt=e[pt]. next)
19
20
              {
                  next=e[pt].to;
21
                      (level[next]!=-1||e[pt].flow <=0) continue;
22
                  level[next] = level[q[low]] + 1;
23
                  q[++high]=next;
24
                      (next=T) return true;
25
26
         return false;
27
   }
28
         Find(int x, int delta)
   int
29
30
         pt, next, temp, res = 0;
   int
31
             (x=T \mid delta <= 0) return delta;
32
         for (pt=last[x]; pt!=-1; last[x]=pt=e[pt]. next)
33
34
              next=e[pt].to;
35
                 (level[next]! = level[x]+1) continue;
36
             temp=Find(next, min(delta, e[pt].flow));
37
              delta-=temp;
38
              res = temp;
39
              e[pt].flow=temp;
40
```

```
e [pt^1]. flow+=temp;

if (delta <=0) return res;

return res;

return res;
```

### 3.3 费用流

```
struct Edge
   {
2
            int to,flow,cost,next;
3
   }
            e[1000050];
   void Add(int x,int y,int flow,int cost)
   {
6
         e[++now]. to=y;
         e [now].flow=flow;
         e[now].cost=cost;
         e [now] . next = first [x];
10
         first[x]=now;
12
   bool Find()
13
   {
14
   int
         low, high, pt, next;
15
         memset(inq, false, sizeof(inq));
16
         memset(f, 127, sizeof(f));
17
         q[low=high=0]=S;
18
         f[S] = 0;
19
         inq[S] = true;
20
         for (; low \le high; inq[q[low++]] = false)
21
              for (pt=first [q[low]]; pt!=-1; pt=e[pt].next)
22
23
                   next=e[pt].to;
24
                        (f[next] \le f[q[low]] + e[pt] \cdot cost ||e[pt]| \cdot flow \le 0||q[low]|
25
                       ==T \mid \text{next} = S) continue;
                   f[next] = f[q[low]] + e[pt].cost;
26
                   path [next]=q[low];
27
                   number [next] = pt;
28
                        (!inq[next])
                   i f
29
30
                        inq[next]=true;
31
                        q[++high]=next;
32
                   }
33
34
         return f[T]<999999999;
35
   }
36
   void Addflow()
37
   {
38
         t = 999999999, i;
   int
39
         for (i=T; i!=S; i=path[i]) t=min(t, e[number[i]]. flow);
40
         mincost+=t*f[T];
41
         \max flow += t;
42
         for (i=T; i!=S; i=path[i])
43
```

# 4 小算法

## 4.1 跨平台大随机数

```
int Rand()
int ra = rand() % 32768;
int rb = rand() % 32768;
return (ra << 15) | rb;
}</pre>
```

### 4.2 读入优化

```
char *st,*nd,ch[1000050];
   inline char Get()
3
                (st=nd)
           i f
                st=ch;
                nd=st+fread (ch,1,1000007, stdin);
           return *(st++);
9
   inline int Read()
11
12
   register int
13
   register char c=Get();
   register bool fu=false;
15
         for (; c!= '- '&&(c<'0' | | '9'<c); c=Get());
16
         i f
              ( c=='-')
17
18
              fu=true;
19
              c=Get();
20
21
         for (; '0' \le c \& c \le '9'; c = Get()) t = 10*t + c - '0';
22
         if (fu) return -t;
23
         return t;
24
25
```

## 4.3 KMP算法

```
len = strlen(st);
```

### 4.4 扩展KMP

```
extend[1] = number;
1
            for (far = 1; far < number; far++)
2
                (tested [far] != tested [far + 1])
                break;
            extend[source = 2] = far - 1;
            for (j = 3; j \le number; j++)
                temp = extend[j - source + 1];
10
                 if (j + temp - 1 < far)
11
12
                    extend[j] = temp;
13
                    else
15
                    for (k = max(far + 1, j); k \le number; k++)
16
                       (\text{tested}[k] != \text{tested}[k - j + 1])
17
18
                        break;
19
20
                    far = k - 1;
21
                    source = j;
22
                    extend[j] = far - source + 1;
23
24
```

# 5 计算几何

### 5.1 凸包

```
1 bool cmp(const Point &a,const Point &b)
2 {
3     return F(a.x-b.x) < 0 | | F(a.x-b.x) == 0 & a.y < b.y;
4 }
5 void Gram(int id[],int n)
6 {
7 int i,mid;
8     sort(id,id+n,cmp);
9     tp=0;</pre>
```

```
//凸包从最小的点出发,逆时针方向x
10
       for (i=0; i< n; i++)
11
12
            for (; tp \ge 2 \& Cross(p[sk[tp-1]] - p[sk[tp-2]], p[id[i]] - p[sk[tp-2]])
13
               -1]) <=0;tp--);
            //有重点必须使用小于等于不留共线点,无重点使用小于等于不留共线点,无重点
14
               使用小于留共线点
            sk[tp++]=id[i];
15
16
       mid=tp;
17
       for (i=n-2; i>=0; i--)
18
19
            for (; tp>mid&&Cross(p[sk[tp-1]]-p[sk[tp-2]], p[id[i]]-p[sk[tp-2]])
20
               -1])<=0;tp--);
            //有重点必须使用小于等于不留共线点,无重点使用小于等于不留共线点,无重点
21
               使用小于留共线点
            sk[tp++]=id[i];
22
23
       i f
           (n>1) tp--;
24
  }
25
```

### 5.2 定义

```
struct Point
1
   {
2
           double x,y;
           Point() {}
           Point (double \_x, double \_y):x(\_x),y(\_y){}
   };
   struct Seg
       Point a,b;
       Seg() {}
10
       Seg(Point _a, Point _b):a(_a),b(_b){}
11
12
   struct Circle
13
   {
14
           double x,y,r;
           Point pt() {return Point(x,y);}
16
           double Area() {return pi*r*r;}
17
   };
18
   Point operator +(const Point &a, const Point &b)
19
20
       \mathbf{return} Point (a.x+b.x,a.y+b.y);
21
22
   Point operator -(const Point &a, const Point &b)
23
24
       return Point(a.x-b.x,a.y-b.y);
25
26
   Point operator *(const Point &a, double b)
27
28
       return Point (a.x*b,a.y*b);
29
30
```

```
Point operator /(const Point &a, double b)
   {
32
        return Point (a.x/b,a.y/b);
33
34
   int F(double x)
35
36
        if (x>eps) return 1;
37
        if (x < -eps) return -1;
38
        return 0;
39
40
   bool operator ==(const Point &a, const Point &b)
41
42
        return F(a.x-b.x) = 0 \& F(a.y-b.y) = 0;
43
44
   double Dist (const Point &a)
45
        return sqrt(a.x*a.x+a.y*a.y);
47
   }
   double Dot(const Point &a, const Point &b)
49
        return a.x*b.x+a.y*b.y;
51
   }
52
   double Cross (const Point &a, const Point &b)
53
54
        return a.x*b.y-a.y*b.x;
55
56
   Point Rotate(const Point &p, double a) // 逆时针旋转
57
58
        return Point (p.x*cos(a)-p.y*sin(a),p.x*sin(a)+p.y*cos(a));
59
60
   Point Inter(Seg a, Seg b) // 两线段相交(前提有交点)
61
62
   double s=Cross(a.b-a.a,b.a-a.a), t=Cross(a.b-a.a,b.b-a.a);
           return b.a+(b.b-b.a)*s/(s-t);
64
   vector<Point> SegCir(Seg seg, Point pt, double r)//线圆
66
67
   vector < Point > ans;
68
   double mul;
69
   Point vec, mid;
70
            ans.clear();
71
            vec=Rotate(seg.b-seg.a, pi/2);
72
            mid=Inter(seg, Seg(pt, pt+vec));
73
            i f
                 (F(Dist(pt-mid)-r)>0) return ans;
74
            i f
                 (F(Dist(pt-mid)-r)==0)
75
76
                      ans.push_back(mid);
77
                      ans.push_back(mid);
78
                      return ans;
79
            vec=seg.b-seg.a;
81
            \text{mul} = \text{sqrt} (r * r - \text{Dist} 2 (\text{mid} - \text{pt})) / \text{Dist} (\text{vec});
82
            ans.push_back(mid+vec*mul);
83
            ans.push_back(mid-vec*mul);
```

```
return ans;
    }
86
    vector < Point > Circir (Circle a, Circle b) //圆圆相交
88
    vector < Point > ans;
89
    double dis, dis2, alpha;
90
    Point pa, pb, vec;
91
           ans.clear();
92
               (a.r < b.r) swap(a,b);
93
           pa=a.pt();
94
           pb=b.pt();
95
           vec=pb-pa;
96
           dis=Dist(vec);
97
           dis2=Dist2 (vec);
98
                (F(dis - (a.r+b.r)) > 0 | |F(dis - (a.r-b.r)) < 0)  return ans;
99
           i f
                (F(dis - (a.r+b.r)) == 0)
100
101
                ans.push_back(pa+vec*a.r/(a.r+b.r));
102
               return ans;
103
104
                (F(dis - (a.r-b.r)) == 0)
           i f
105
106
                ans.push_back(pa+vec*a.r/(a.r-b.r));
107
               return ans;
108
109
           alpha=acos((a.r*a.r+dis2-b.r*b.r)/2/a.r/dis);
110
           ans.push_back(pa+Rotate(vec, alpha)*a.r/dis);
111
           ans.push_back(pa+Rotate(vec,-alpha)*a.r/dis);
112
           return ans;
113
    }
114
    double Bing (double ra, double rb, double dis)
115
116
    double alpha, beta;
117
            i f
                 (ra < rb) swap(ra, rb);
118
                 (F(dis-(ra-rb)) \le 0) return pi*ra*ra;
            i f
119
                 (F(dis-(ra+rb)))>=0) return pi*ra*ra+pi*rb*rb;
120
            alpha=acos((ra*ra+dis*dis-rb*rb)/2/dis/ra);
121
            beta=acos((rb*rb+dis*dis-ra*ra)/2/dis/rb);
122
            return (pi-alpha)*ra*ra+(pi-beta)*rb*rb+ra*dis*sin(alpha);
123
    }
124
125
    double Jiao (double ra, double rb, double dis)
126
127
            return pi*ra*ra+pi*rb*rb-Bing(ra,rb,dis);
128
    }
129
130
    Point Gongmid(Circle a, Circle b) //正确性待定
131
132
    Point pa=a.pt(),pb=b.pt();
133
           return pa+(pb-pa)*a.r/(a.r+b.r);
134
    }
135
136
    Point Gongright (Circle a, Circle b)
137
    {
138
```

```
Point pa=a.pt(),pb=b.pt();
           return pa+(pb-pa)*a.r/(a.r-b.r);
140
141
142
    int Ptinpol (Point pt)
143
144
    int i, k, d1, d2, wn=0;
145
        for (i = 0; i < n; i++)
146
147
             if (Ins (pt, Seg (p[i], p[(i+1)\%n]))) return 2;
148
             k=F(Cross(p[(i+1)\%n]-p[i],pt-p[i]));
149
             d1=F(p[i].y-pt.y);
150
             d2 = F(p[(i+1)\%n].v-pt.v);
151
             if(k>0\&\&d1<=0\&\&d2>0)wn++;
152
             if(k<0\&\&d2<=0\&\&d1>0)wn--;
153
154
        return wn!=0;
155
156
    bool Cirinpol(Point pt)//需要点在多边形内的前提
157
    int
         i ;
159
    double nearest;
160
         nearest=1e+100;
161
         for (i=0;i< n;i++)
162
163
              nearest=min(nearest, Dist(p[i]-pt));
164
                   (F(Dot(pt-p[i], p[(i+1)\%n]-p[i]))>0&&
165
                   F(Dot(pt-p[(i+1)\%n], p[i]-p[(i+1)\%n]))>0)
166
                   nearest=min(nearest, abs(Cross(p[i]-pt,p[(i+1)%n]-pt))/
167
                       dis[i]);
168
         return F(nearest-r)>=0;
169
170
    bool Ins (const Point &p, const Seg &s)
171
    {
172
         return F(Cross(s.a-p,s.b-p))==0&&
173
                  F(p.x-min(s.a.x,s.b.x)) > = 0 \& \&
174
                  F(p.x-max(s.a.x,s.b.x)) \le 0
175
                  F(p.y-min(s.a.y,s.b.y)) >= 0 \& \&
176
                  F(p.y-max(s.a.y,s.b.y)) <=0;
177
178
    double PS(const Point &p, const Seg &s) 点到线段最短距离
179
180
                  (F(Dot(p-s.a, s.b-s.a)) < 0 | | F(Dot(p-s.b, s.a-s.b)) < 0) return
181
                  \min(\text{Dist}(p-s.a), \text{Dist}(p-s.b));
             return abs(Cross(s.a-p,s.b-p))/Dist(s.a-s.b);
182
183
    double SS(const Seg &a, const Seg &b) 线段到线段最短距离
184
185
            return \min(\min(PS(a.a,b),PS(a.b,b)),\min(PS(b.a,a),PS(b.b,a)));
187
    double Alpha (Point a, Point b)
188
189
    double ans;
190
```

```
ans=atan2(b.y,b.x)-atan2(a.y,a.x);
191
                 (ans < 0) ans=-ans:
192
                 (ans>pi) ans=2*pi-ans;
            i f
193
            return ans;
194
195
    double Shan (Circle c, double a)
196
    {
197
            return c.r*c.r*a/2;
198
199
```

### 5.3 半平面交

```
1
   bool Cmphp(Seg a, Seg b)
2
   Point va, vb;
   double dega, degb;
         va=a.b-a.a;
         vb=b.b-b.a;
         dega=atan2(va.y,va.x);
         degb=atan2(vb.y,vb.x);
         return F(dega-degb) < 0 || F(dega-degb) = 0 \& Cross(a.b-a.a,b.a-a.a)
10
             < 0;
11
   void HalfPlane (Seg hp[], int n, Point pol[], int &pols)
   {
13
   int
         tp, i, low, high;
14
   Point mid;
15
         hp[n++]=Seg(Point(-oo,-oo),Point(oo,-oo));
16
         hp[n++]=Seg(Point(oo,-oo),Point(oo,oo));
17
         hp[n++]=Seg(Point(oo,oo),Point(-oo,oo));
18
         hp[n++]=Seg(Point(-oo,oo),Point(-oo,-oo));
19
         sort (hp, hp+n, Cmphp);
20
         tp=0;//sk \ \theta^*tp-1
21
         low = 0;
22
         high=-1;
23
         for (i=0;i< n;i++)
24
             (high-low+1==0||F(Cross(sk[high].b-sk[high].a,hp[i].b-hp[i].
25
            a)))
26
             for (; low<high; high--)
27
                  mid=Inter(sk[high], sk[high-1]);
29
                      (F(Cross(hp[i].b-hp[i].a,mid-hp[i].a))>0) break;
30
31
             for (; low<high; low++)</pre>
32
33
                  mid=Inter(sk[low], sk[low+1]);
34
                      (F(Cross(hp[i].b-hp[i].a,mid-hp[i].a))>0) break;
35
36
             sk[++high]=hp[i];
37
38
```

```
for (; low<high; high--)
40
                   mid=Inter(sk[high], sk[high-1]);
41
                         (\operatorname{Cross}(\operatorname{sk}[\operatorname{low}].\operatorname{b-sk}[\operatorname{low}].a,\operatorname{mid-sk}[\operatorname{low}].a)>0) break;
42
43
            tp = high - low + 1;
44
            for (i=0; i < tp; i++) sk [i] = sk [low+i];
45
             pols = 0:
46
                   (tp \le 2) return;
            i f
47
            for (i=0; i < tp; i++) pol [pols++]=Inter(sk[i], sk[(i+1)\%tp]);
48
49
    }
```

### 5.4 圆与多边形交集

```
double CT(Circle c, Point a, Point b) 圆与三角形交(多边形)
   {
2
   double da, db;
3
   Seg
            s;
   vector <Point> temp;
            da=Dist(a-c.pt());
           db=Dist(b-c.pt());
                (da>db)
                swap(a,b);
10
                swap (da, db);
11
12
            s=Seg(a,b);
13
            temp=CS(c, s);
14
                (F(db-c.r) \le 0) return 0.5*abs(Cross(a-c.pt(),b-c.pt()));
            i f
15
            i f
                (F(da-c.r)<0)
16
17
                     (F(Dot(a-temp[1],b-temp[1]))<0) swap(temp[0],temp[1]);
18
                return Shan(c, Alpha(temp[0] - c.pt(), b-c.pt())) + 0.5*abs(
19
                     Cross(a-c.pt(),temp[0]-c.pt());
            i f
                (!temp.size()) return Shan(c, Alpha(a-c.pt(),b-c.pt()));
21
                (Ins(temp[1], s)\&\&Dist2(a-temp[1]) < Dist2(a-temp[0])) swap(
            i f
22
                temp[0], temp[1]);
                (\operatorname{Ins}(\operatorname{temp}[0], s) \&\& \operatorname{Ins}(\operatorname{temp}[1], s))
23
            i f
            {
24
                return Shan (c, Alpha (a-c.pt(), temp[0] - c.pt()))+
25
                         Shan(c, Alpha(b-c, pt(), temp[1]-c, pt()))+
26
                         0.5*abs(Cross(temp[0]-c.pt(),temp[1]-c.pt()));
27
28
            return Shan(c, Alpha(a-c.pt(),b-c.pt()));
29
30
```

### 5.5 三角形面积并

```
| #include<math.h>
| #include<stdio.h>
```

```
#include < string.h>
   #include < algorithm >
  #define N 333
  #define pr pair<ld,ld>
   using namespace std;
   typedef long double ld;
   const ld EPS=1e-8;
   const ld INF=1e100;
   struct Point
11
   {
12
        ld x, y;
13
        Point(){}
14
        Point (ld_-, ld_-) : x(_), y(_-) \{ \}
15
        void read()
16
17
            double _ , _ _ ;
            scanf("%lf%lf",&_-,&_-);
19
            x =_{-}, y =_{--};
21
        friend bool operator <(Point a, Point b)
23
            if(fabs(a.x-b.x) < EPS)
24
            return a.y<br/>b.y;
25
            return a.x<b.x;
26
27
        friend Point operator +(Point a, Point b)
28
29
            return Point (a.x+b.x, a.y+b.y);
30
31
        friend Point operator -(Point a, Point b)
32
33
            return Point (a.x-b.x,a.y-b.y);
34
35
        friend Point operator *(ld a, Point b)
36
            return Point(a*b.x,a*b.y);
38
39
        friend ld operator *(Point a, Point b)
40
41
            return a.x*b.x+a.y*b.y;
42
43
        friend ld operator ^(Point a, Point b)
44
45
            return a.x*b.y-a.y*b.x;
46
47
   a[N][3], Poi[N*N];
48
   struct Line
49
   {
50
        Point p, v;
51
        Line() {}
52
        Line(Point_{-}, Point_{-}) \{p=_{-}, v=_{--};\}
53
        Point operator [](int k)
54
55
            if (k)
                     return p+v;
```

```
else
                       return p;
57
58
         friend bool Cross (Line a, Line b)
59
60
             return (a.v^b[0]-a.p)*(a.v^b[1]-a.p)<-EPS&&(b.v^a[0]-b.p)*(b.
61
                 v^a[1] - b.p < -EPS;
62
         friend Point getP(Line a, Line b)
63
64
             Point u=a.p-b.p;
65
             1d \text{ temp}=(b.v^u)/(a.v^b.v);
66
             return a.p+temp*a.v;
67
68
    1[N][3],T;
69
    pr p[N];
70
    int main()
71
    {
72
        int n,m,i,j,k,x,y,cnt,tot;
73
        ld ans, last, A, B, sum;
74
         scanf("%d",&n);
75
        for (i=1, tot=0; i \le n; i++)
76
77
             a[i][0].read(),a[i][1].read(),a[i][2].read();
78
             Poi[++tot]=a[i][0], Poi[++tot]=a[i][1], Poi[++tot]=a[i][2];
79
             sort(a[i], a[i]+3);
80
             if ((a[i][2] - a[i][0] ^ a[i][1] - a[i][0])>EPS)
81
                  l[i][0] = Line(a[i][0], a[i][2]), l[i][1] = Line(a[i][2], a[i][2])
82
                      [1], [1], [i]
             else
83
                  l[i][0] = Line(a[i][2], a[i][0]), l[i][1] = Line(a[i][1], a[i
84
                      [2]), [1] [2] = Line (a[i][0], a[i][1]);
85
        for (i=1; i \le n; i++)
86
87
             for (j=1; j< i; j++)
89
                  for (x=0;x<3;x++)
                       for (y=0;y<3;y++)
91
92
                            if (Cross(l[i][x],l[j][y]))
93
                                Poi[++tot] = getP(1[i][x], 1[j][y]);
94
                       }
95
96
97
         sort(Poi+1,Poi+tot+1);
98
        ans=0, last=Poi[1].x;
99
        T=Line(Point(0,-INF),Point(0,INF));
100
        for(i=2;i<=tot;i++)
101
102
             T.p. x = (last + Poi[i].x)/2;
103
             for (j=1, cnt=0; j \le n; j++)
104
105
                  if (Cross (1 [ j ] [ 0 ] , T ) )
106
107
```

```
A=getP(1[j][0],T).y;
108
                       if (Cross (l[j][1],T))
109
                            B=getP(l[j][1],T).y;
110
                       else
111
                            B=getP(1[j][2],T).y;
112
                       if(A>B) \quad swap(A,B);
113
                       p[++cnt]=pr(A,B);
114
115
116
              sort(p+1,p+cnt+1);
117
             for (j=1,sum=0,A=-INF; j<=cnt; j++)
118
119
                   if(p[j].first>A)
120
121
                       sum+=p[j].second-p[j].first;
122
                       A=p[j]. second;
123
124
                   else
125
126
                       if(p[j].second>A)
127
                            sum+=p[j].second-A,A=p[j].second;
128
129
130
              ans+=(Poi[i].x-last)*sum;
131
             last=Poi[i].x;
132
133
         printf("\%.2lf\n",(double)ans);
134
         return 0;
135
136
```

### 5.6 K圆并

```
#include <cstdio>
  #include <cstdlib>
  #include <climits>
4 #include <iostream>
5 #include <algorithm>
  #include <cstring>
  #include <string>
  #include <queue>
  #include <map>
  #include <vector>
  #include <bitset>
  #include <cmath>
  #include <set>
  #include <utility>
  #include <ctime>
  #define sqr(x) ((x)*(x))
16
  using namespace std;
17
18
  const int N = 1010;
  const double eps = 1e-8;
```

```
const double pi = acos(-1.0);
   double area [N];
22
   int n;
24
   int dcmp(double x) {
25
        if (x < -eps) return -1; else return x > eps;
26
   }
27
28
   struct cp {
29
        \mathbf{double} \ x\,,\ y\,,\ r\,,\ \mathrm{angle}\,;
30
        int d;
31
        cp(){}
32
        cp(double xx, double yy, double ang = 0, int t = 0) {
33
            x = xx; y = yy; angle = ang; d = t;
34
35
        void get() {
36
            scanf("%1f%1f%1f", &x, &y, &r);
37
            d = 1;
39
   cir[N], tp[N * 2];
40
41
   double dis(cp a, cp b) {
42
        return \operatorname{sqrt}(\operatorname{sqr}(a.x - b.x) + \operatorname{sqr}(a.y - b.y));
43
44
45
   double cross (cp p0, cp p1, cp p2) {
46
        return (p1.x - p0.x) * (p2.y - p0.y) - (p1.y - p0.y) * (p2.x - p0)
47
            .x);
   }
48
49
   int CirCrossCir(cp p1, double r1, cp p2, double r2, cp &cp1, cp &cp2)
50
        double mx = p2.x - p1.x, sx = p2.x + p1.x, mx2 = mx * mx;
51
        double my = p2.y - p1.y, sy = p2.y + p1.y, my2 = my * my;
52
        double sq = mx^2 + my^2, d = -(sq - sqr(r^1 - r^2)) * (sq - sqr(r^1 + r^2))
53
           r2));
        if (d + eps < 0) return 0; if (d < eps) d = 0; else d = sqrt(d);
        double x = mx * ((r1 + r2) * (r1 - r2) + mx * sx) + sx * my2;
55
        double y = my * ((r1 + r2) * (r1 - r2) + my * sy) + sy * mx2;
        \mathbf{double} \ dx = mx * d, \ dy = my * d; \ sq *= 2;
57
        cp1.x = (x - dy) / sq; cp1.y = (y + dx) / sq;
58
        cp2.x = (x + dy) / sq; cp2.y = (y - dx) / sq;
59
        if (d > eps) return 2; else return 1;
60
   }
61
62
   bool circmp (const cp& u, const cp& v) {
63
        return dcmp(u.r - v.r) < 0;
64
   }
65
66
   bool cmp(const cp& u, const cp& v) {
67
        if (dcmp(u.angle - v.angle)) return u.angle < v.angle;</pre>
68
        return u.d > v.d;
69
   }
70
71
```

```
double calc(cp cir, cp cp1, cp cp2) {
        double ans = (cp2.angle - cp1.angle) * sqr(cir.r)
73
             -\operatorname{cross}(\operatorname{cir}, \operatorname{cp1}, \operatorname{cp2}) + \operatorname{cross}(\operatorname{cp}(0, 0), \operatorname{cp1}, \operatorname{cp2});
74
        return ans / 2;
75
    }
76
77
    void CirUnion(cp cir[], int n) {
78
        cp cp1, cp2;
79
         sort(cir, cir + n, circmp);
80
        for (int i = 0; i < n; ++i)
81
             for (int j = i + 1; j < n; ++j)
82
                  if (dcmp(dis(cir[i], cir[j]) + cir[i].r - cir[j].r) \le 0)
83
                       cir [i].d++;
84
        for (int i = 0; i < n; ++i) {
85
             int tn = 0, cnt = 0;
86
             for (int j = 0; j < n; ++j) {
                  if (i == j) continue;
88
                  if (CirCrossCir(cir[i], cir[i].r, cir[j], cir[j].r,
                       cp2, cp1) < 2) continue;
90
                  cp1.angle = atan2(cp1.y - cir[i].y, cp1.x - cir[i].x);
91
                  cp2.angle = atan2(cp2.y - cir[i].y, cp2.x - cir[i].x);
92
                  cp1.d = 1;
                                  tp[tn++] = cp1;
93
                  cp2.d = -1;
                                  tp[tn++] = cp2;
94
                  if (dcmp(cp1.angle - cp2.angle) > 0) cnt++;
95
96
             tp[tn++] = cp(cir[i].x - cir[i].r, cir[i].y, pi, -cnt);
97
             tp[tn++] = cp(cir[i].x - cir[i].r, cir[i].y, -pi, cnt);
98
             sort(tp, tp + tn, cmp);
99
             int p, s = cir[i].d + tp[0].d;
100
             for (int j = 1; j < tn; ++j) {
101
                  p = s; s += tp[j].d;
102
                  area[p] += calc(cir[i], tp[j-1], tp[j]);
103
104
        }
105
    }
106
107
    void solve() {
108
        for (int i = 0; i < n; ++i)
109
             cir[i].get();
110
        memset(area, 0, sizeof(area));
111
        CirUnion(cir, n);
112
         for (int i = 1; i \le n; ++i) {
113
             area[i] -= area[i + 1];
114
             printf("[\%d] = \ \%.31f \ n", i, area[i]);
115
         }
116
    }
117
118
    int main() {
119
         freopen ("a.in", "r", stdin);
120
        while (scanf("%d", &n) != EOF) {
121
             solve();
122
123
        return 0;
124
125
```

## 6 三维计算几何

### 6.1 基本定义

```
Point Cross (Point a, Point b)
2
          return Point (a.y*b.z-a.z*b.y, a.z*b.x-a.x*b.z, a.x*b.y-a.y*b.x);
3
4
   double Crossxy (Point a, Point b)
6
           return a.x*b.y-a.y*b.x;
   vector < Point > SegPlane (Seg seg, Plane p)
10
   double
                  s, t;
11
   Point
                   fa;
12
   vector < Point > ans;
                  ans.clear();
14
                   fa=Cross(p.b-p.a,p.c-p.a);
15
                      (F(Dot(fa, seg.b-seg.a))==0) return ans;
16
                   s=Dot(p.a-seg.a,fa)/Dist(fa);
17
                  t=Dot(p.a-seg.b,fa)/Dist(fa);
18
                  ans.push_back(seg.a+(seg.b-seg.a)*s/(s-t));
19
                  return ans;
20
21
```

#### 6.2 一些补充

```
\\ mixed product
   double Mix(Point3 a, Point3 b, Point3 c)
2
3
           return Dot(Cross(a,b),c);
   \\ distance from point to plane
   double PP(Point3 pt, Plane pl)
   Point3 fa=Cross(pl.b-pl.a,pl.c-pl.a);
           return abs(Dot(fa,pt-pl.a))/Dist(fa);
10
11
   \\ get the center point from 3D(need plane well prepared)
12
   Point3 Getcenter (Point3 p[], int n, Plane pp[], int nn)
13
14
   int i;
15
   double sumv, tempv;
16
   Point3 sum;
17
           sumv = 0:
           sum = Point3(0,0,0);
19
           for (i=0; i< nn; i++)
20
21
                    tempv=Mix(pp[i].b-pp[i].a,pp[i].c-pp[i].a,Point3
22
                        (0,0,0)-pp[i].a);
                    sum=sum+(pp[i].a+pp[i].b+pp[i].c)*tempv/4.0;
23
```

## 7 bitset用法

```
C++ bitset 用法
  C的++ bitset 在 bitset 头文件中,它是一种类似数组的结构,它的每一个元素只能是0
     或1,每个元素仅用1空间。bit下面是具体用法构造函数常用构造函数有四种,如下
6
  bitset 复制代码
10
11
                        ; //无参构造,长度为4,默认每一位为0
      bitset <4> bitset1
12
13
                            ; //长度为8,二进制保存,前面用0补充
      bitset < 8 > bitset 2 (12)
15
      string \ s = "100101";
16
      bitset < 10 > bitset 3 (s)
                            ;//长度为,前面用0补充10
17
      char s2[] = "10101";
19
      bitset < 13 > bitset 4 (s2)
                             ;//长度为,前面用0补充13
20
21
      cout << bitset1 << endl
                             ;//0000
22
      cout << bitset2 << endl
                              ;//00001100
23
                             ;//0000100101
      cout << bitset3 << endl
24
      cout << bitset4 << endl
                              ;//00000001010101复制代码注意:用字符串构造
25
         时,字符串只能包含
26
27
29
   '0' 或 '1', 否则会抛出异常。构造时, 需在中表明
31
  <⇒bitset 的大小即(size)。在进行有参构造时,若参数的二进制表示比的小,则在前面用
33
34
  bitsetsize如上面的栗子();若比大,参数为整数时取后面部分,参数为字符串时取前面部
     分bitsize如下面栗子(): 复制代码
36
37
                           ;//的二进制为(长度为4),
      bitset <2> bitset1(12)
38
         但121100的bitset1size, 只取后面部分, 即=200
39
      string \ s = "100101"
40
```

```
;//的ssize,而=6的bitsetsize,只取前面部分,
        bitset < 4 > bitset 2 (s)
41
           即 =41001
42
        char s2 [] = "11101";
43
        bitset < 4 > bitset 3 (s2)
                                   ;//与同理, 只取前面部分, 即 bitset21110
44
45
        cout << bitset1 << endl
                                     ;//00
46
                                     ;//1001
        \operatorname{cout} << \operatorname{bitset2} << \operatorname{endl}
47
                                     ;//1110复制代码可用的操作符对于二进制有位操作
        cout << bitset3 << endl
48
           符, 具体如下
49
50
51
   bitset 复制代码
53
55
        bitset <4> foo (string ("1001"));
        bitset <4> bar (string ("0011"));
57
        cout << (foo^=bar) << endl;
                                             // 1010 (对按位异或后赋值
59
           给foobarfoo)
        cout << (foo&=bar) << endl;
                                             // 0010 按位与后赋值给(foo)
60
                                             // 0011 按位或后赋值给(foo)
        cout \ll (foo|=bar) \ll endl;
61
62
                                             // 1100 左移 2 位, 低位补 0, 有自身赋
        cout \ll (foo \ll 2) \ll endl;
63
        cout \ll (foo >>=1) \ll endl;
                                              // 0110 右移 1 位,高位补 0 ,有自身赋 ||
64
           值()
65
                                              // 1100 按位取反()
        cout << (~bar) << endl;
66
                                             // 0110 左移, 不赋值()
        cout \ll (bar \ll 1) \ll endl;
67
        cout \ll (bar >> 1) \ll endl;
                                             // 0001 右移,不赋值()
68
        \mathrm{cout} \, << \, (\, \mathrm{foo} \underline{\hspace{-0.05in}} \mathrm{bar} \,) \, << \, \mathrm{endl} \, ;
                                             // false 为(0110==0011false)
70
        cout \ll (foo!=bar) \ll endl;
                                             // true 为 (0110!=0011true)
71
72
                                             // 0010 按位与,不赋值()
        cout << (foo&bar) << endl;</pre>
73
                                             // 0111 按位或,不赋值()
        cout << (foo|bar) << endl;</pre>
74
                                             // 0101 按位异或, 不赋值()复制代码
        cout << (foo^bar) << endl;</pre>
75
           此外,可以通过
76
     [ ] 访问元素类似数组(),注意最低位下标为0,如下:
77
78
        bitset <4> foo ("1011");
79
80
        cout \ll foo[0] \ll endl
                                    ; //1
81
        cout \ll foo[1] \ll endl
                                    ; //1
82
                                    ;//0当然,通过这种方式对某一位元素赋值也是可以
        cout \ll foo[2] \ll endl
83
           的,栗子就不放了。可用函数还支持一些有意思的函数,比如:
85
86
87
```

```
bitset复制代码
 90
        bitset <8> foo ("10011011");
92
 93
        cout << foo.count() << endl</pre>
                                             (5函数用来求中的位数,中共有5个
                                      ;//
94
           1 countbitset1foo
        cout << foo.size() << endl
                                             (8函数用来求的大小,一共有8
                                     ; //
95
           位。sizebitset
96
        cout \ll foo.test(0) \ll endl
                                      ;//
                                             (函数用来查下标处的元素是 0 还是
97
           1, 并返回或, 此处 true test false true foo 为 1, 返回 [0] true
        cout \ll foo.test(2) \ll endl
                                      ;//
                                             (同理, falsefoo为 0, 返
98
           \Box /2/false
 99
                                           (函数检查中是否有 1 trueanybitset
        cout << foo.any() << endl
                                    ; //
100
                                            (函数检查中是否没有
        cout << foo.none() << endl</pre>
                                     ;//
101
           1 falsenonebitset
                                           (函数检查中是全部为
        cout << foo.all() << endl
102
           1 falseallbitset复制代码补充说明一下:函数会对下标越界作出检查,而通过
103
104
    test [ ] 访问元素却不会经过下标检查,所以,在两种方式通用的情况下,选择函数更安全
105
       一些test另外,含有一些函数:
                                 复制代码
106
107
108
109
110
111
        bitset <8> foo ("10011011");
112
113
        cout \ll foo.flip(2) \ll endl
                                      ; //
                                             (10011111函数传参数时,用于将参
114
           数位取反,本行代码将下标 2 \, \text{处} \, flipfoo 反转,即 0 \, \text{变} \, 1 , 1 \, \text{变} \, 0 ""
        cout << foo.flip() << endl
                                             (01100000函数不指定参数时,将每
                                    ; //
115
           一位全部取反 flip bitset
116
        cout << foo.set() << endl
                                               (111111111) 函数不指定参数时,将
                                        ;//
117
           的每一位全部置为 1 setbitset
        \operatorname{cout} << \operatorname{foo.set}(3,0) << \operatorname{endl}
                                              (11110111函数指定两位参数时,
118
           将第一参数位的元素置为第二参数的值,本行对的操作相当于setfoofoo[3]=0
        cout \ll foo.set(3) \ll endl
                                              (11111111)函数只有一个参数时,
119
           将参数下标处置为 1 set
120
                                              (11101111)函数传一个参数时将参
        cout \ll foo.reset(4) \ll endl
                                       ;//
121
           数下标处置为 0 reset
                                              (000000000函数不传参数时将的每
        cout << foo.reset() << endl
122
           一位全部置为 0 resetbitset复制代码同样,它们也都会检查下标是否越界,如果
           越界就会抛出异常最后,还有一些类型转换的函数,如下:复制代码
123
124
125
126
127
128
        bitset <8> foo ("10011011");
129
130
```

```
string s = foo.to_string() ;//将转换成类型 bitsetstring unsigned long a = foo.to_ulong() ;//将转换成 bitsetunsigned 类
131
132
           型 long
unsigned long long b = foo.to_ullong()
                                                                        ; //将转换
133
                成bitsetunsigned long 类型long
134
                                         ;//10011011
;//155
           cout <\!< s <\!< endl
135
           \mathrm{cout} \,<\!<\, \mathrm{a} \,<\!<\, \mathrm{endl}
136
                                          ;//155复制代码
           cout <\!< b <\!< endl
137
```