Andrew算法求凸包

```
int n;
const int maxn=1e5+5;
struct Point{
   double x,y;
};
Point p[maxn],ch[maxn];    //后者记录凸包上的点

bool cmp(Point x,Point y)
{
    return x.x<y.x||(x.x==y.x&&x.y<y.y);//x从小到大排序，如果x相同则y从小到大排序
}
int Cross(Point x,Point y,Point z)
{
    double x1=x.x-y.x;
    double y1=x.y-y.y;
    double x2=z.x-y.x;
    double y2=z.y-y.y;
    if((x1*y2-x2*y1)<=0) return 0;//如果不希望在凸包的边上有输入点。把<=改成<
    return 1;
}
void andrew()
{
    scanf("%d",&n);
    for(int i=1;i<=n;i++)
        scanf("%lf %lf",&p[i].x,&p[i].y);
    sort(p+1,p+n+1,cmp);
    int m=1;
    int i;
    for(i=1;i<=n;i++)//从左到右扫描
    {
        while( m>2 && !Cross(ch[m-1],ch[m-2],p[i])) m--;
        ch[m++]=p[i];
    }
    int k=m;
    for(i=n-1;i>0;i--)//从右到左扫描
    {
        while( m>k && !Cross(ch[m-1],ch[m-2],p[i])) m--;
        ch[m++]=p[i];
    }
    if(n>2) m--;//凸包有m个顶点
    int ans=m;
    ch[ans+1]=ch[1];
    double sum=0;//周长
    for(int i=1;i<=ans;i++){
        sum+=sqrt((ch[i+1].x-ch[i].x)*(ch[i+1].x-ch[i].x)+(ch[i+1].y-ch[i].y)*
(ch[i+1].y-ch[i].y)); //算周长
```

```
    }
    printf("%.2lf",sum);
}
```

旋转卡壳模版

```
#include<bits/stdc++.h>
#define ll long long
#define inf 1e18
#define mn 100005
using namespace std;
const double dinf=12345678910,eps=1e-10,pi=acos(-1);
struct point{
    double x,y;
    point(double x=0,double y=0):x(x),y(y){ }//构造函数
}a[mn];
typedef point vect;
vect operator + (vect A,vect B){return vect(A.x+B.x,A.y+B.y);}
vect operator - (point A,point B){return vect(A.x-B.x,A.y-B.y);}
vect operator * (vect A,double p){return vect(A.x*p,A.y*p);}
vect operator / (vect A,double p){return vect(A.x/p,A.y/p);}
bool operator < (const point& a,const point& b) {return a.x<b.x||
(a.x==b.x&&a.y<b.y);}
int sgn(double x){if(fabs(x)<eps)return 0;else return x<0?-1:1;}//求一个小数的符号
bool operator == (const point& a, const point& b){return !sgn(a.x - b.x) &&
!sgn(a.y - b.y);}
double polar_angle(vect A){return atan2(A.y,A.x);}//求向量的倾斜角
inline double d_to_r(double D){return pi/180*D;}//角度转弧度
double cross(vect A,vect B){return A.x*B.y-B.x*A.y;}//求两个向量的叉积
double side(point a,point b,point c){return cross(b-a,c-a);}//为正则表示点c(ac向量)在
ab向量的逆时针方向
double dis(point a,point b){return sqrt((a.x-b.x)*(a.x-b.x)+(a.y-b.y)*(a.y-
b.y));}//两点间距离
vect rotate(vect A,double rad)//计算向量逆时针旋转rad后的向量
{return vect(A.x*cos(rad)-A.y*sin(rad),A.x*sin(rad)+A.y*cos(rad));}
ll n,m,s[mn],top,ans=0;
void andrew()
{ll i,tmp;
        sort(a+1,a+n+1);
        s[1]=1;s[2]=2;
        top=2;
        for(i=3;i<=n;i++)
        {
                while(top>1&&side(a[s[top-1]],a[s[top]],a[i])<=0)top--;
                s[++top]=i;
        }
        tmp=top;
        for(i=n-1;i>=1;i--)
        {
                while(top>tmp&&side(a[s[top-1]],a[s[top]],a[i])<=0)top--;
                s[++top]=i;
        }
```

```
}
ll area(point a,point b,point c){return cross(b-a,c-a);}//用叉积计算三角形面积的2倍
ll Dis(point a,point b){return (a.x-b.x)*(a.x-b.x)+(a.y-b.y)*(a.y-b.y);}//两点间距离
的平方
void getlen()//最后计算的是凸包直径的平方
{ll i,j=3;
        if(top==3){ans=Dis(a[s[1]],a[s[2]]);return ;}
        for(i=1;i<top;i++)
        {
                while(area(a[s[i]],a[s[i+1]],a[s[j]])
<area(a[s[i]],a[s[i+1]],a[s[j+1]]))j=j%(top-1)+1;
                ans=max(ans,max(Dis(a[s[i]],a[s[j]]),Dis(a[s[i+1]],a[s[j]])));
        }
}
int main()
{
        ll x,y,z,i,j,k;
        char ch;
        cin>>n;
        for(i=1;i<=n;i++)
                scanf("%lf%lf",&a[i].x,&a[i].y);
        andrew();
        getlen();
        cout<<ans<<endl;
        return 0;
}
```