```
//N 条直线交点多少个
#include<iostream>
using namespace std;
struct node
{
	double x1,y1,x2,y2;
	int cnt;//第 cnt 条线段
}line[40000];
int cnt=0;
int n;
double direction(double x,double y,double x1,double y1,double x2,double y2)   //叉积
{
	double a1=x1-x;
	double b1=y1-y;
	double a2=x2-x;
	double b2=y2-y;
	return a1*b2-a2*b1;
}
int on_segment(double x1,double y1,double x2,double y2,double x,double y)   //判断共线
{
	if((min(x1,x2)<=x && x<=max(x1,x2)) && (min(y1,y2)<=y && y<=max(y1,y2)))   return
1;
	return 0;
}
int cross(int v,int t)//是否交叉，根据算法导论写的
{
	double d1,d2,d3,d4;
	d1=direction(line[t].x1,line[t].y1,line[t].x2,line[t].y2,line[v].x1,line[v].y1);      //算叉积
	d2=direction(line[t].x1,line[t].y1,line[t].x2,line[t].y2,line[v].x2,line[v].y2);
	d3=direction(line[v].x1,line[v].y1,line[v].x2,line[v].y2,line[t].x1,line[t].y1);
	d4=direction(line[v].x1,line[v].y1,line[v].x2,line[v].y2,line[t].x2,line[t].y2);
	if(d1*d2<0 && d3*d4<0) return 1;
	if(!d1 && on_segment(line[t].x1,line[t].y1,line[t].x2,line[t].y2,line[v].x1,line[v].y1)) return 1;
	if(!d2 && on_segment(line[t].x1,line[t].y1,line[t].x2,line[t].y2,line[v].x2,line[v].y2)) return 1;
	if(!d3 && on_segment(line[v].x1,line[v].y1,line[v].x2,line[v].y2,line[t].x1,line[t].y1)) return 1;
	if(!d4 && on_segment(line[v].x1,line[v].y1,line[v].x2,line[v].y2,line[t].x2,line[t].y2)) return 1;
	return 0;
}
int main()
{
	int i,j,sum=0;
	cin>>n;
	while(n--)
	{
```

```
            cin>>line[cnt].x1>>line[cnt].y1>>line[cnt].x2>>line[cnt].y2;
            for(i=0;i<cnt;i++)
            {
                if(cross(i,cnt)==1) sum++;
            }
            cnt++;
        }
        cout<<sum<<endl;
        return 0;
}
```

凸包周长

```
#include <iostream>
#include <cmath>
#include <cstdio>
#include <cstdlib>
using namespace std;
struct point
{
    double x;
    double y;
}p[300000],stack[300000];
int top=2;
double crossProd(point A,point B,point C) //极角
{
    return (B.x-A.x)*(C.y-A.y)-(B.y-A.y)*(C.x-A.x);
}
//以最左下的点为基准点，其他各点（逆时针方向）以极角从小到大的排序规则
double dis(point A,point B)
{
    return sqrt((A.x-B.x)*(A.x-B.x)+(A.y-B.y)*(A.y-B.y));
}
bool iszero(double x)
{
    return (x>0?x:-x)<1e-8;
}
int cmp(const void *a, const void *b)
{
    point *c=(point*)a;
    point *d=(point*)b;
    double k=crossProd(p[0],*c,*d);//极角大小转化为求叉乘
    if (k<1e-8||iszero(k)&&dis(p[0],*c)>dis(p[0],*d)) return 1;
    return -1;
```

```
}
void Graham(int n)
{
    int x,y,min=0,i;
    x=p[0].x;
    y=p[0].y;
    for(i=1;i<n;i++) //找到最左下的一个点
    {
        if(p[i].x<x||(p[i].x==x&&p[i].y<y))
        {
            x=p[i].x;
            y=p[i].y;
            min=i;
        }
    }
    point tmp=p[min];
    p[min]=p[0];
    p[0]=tmp;
    qsort(p+1,n-1,sizeof(point),cmp);
    p[n]=p[0];
    stack[0]=p[0];
    stack[1]=p[1];
    stack[2]=p[2];
    for(i=3;i<=n;i++) //加入一个点后,向右偏拐或共线，则上一个点不在凸包内，则--top，该过程直到不向右偏拐或没有三点共线的点
    {
        while(crossProd(stack[top-1],stack[top],p[i])<=1e-8&&top>=2) top--;
        stack[++top]=p[i];//在当前情况下符合凸包的点，入栈
    }
    return;
}

int main()
{
    int n,i;
    double len;
    cin>>n;
    for(i=0;i<n;i++) cin>>p[i].x>>p[i].y;
    Graham(n);
    for(i=0;i<top;i++)                    len=len+sqrt((stack[i].x-stack[i+1].x)*(stack[i].x-stack[i+1].x)+(stack[i].y-stack[i+1].y)*(stack[i].y-stack[i+1].y));
    printf("%.2lf\n",len);
    return 0;
}
```

点集中任意两点最大距离

```
#include <cstdio>
#include <cmath>
#include <algorithm>
using namespace std;
struct P{
	double x,y;
}p[200000],q[200000];
int n,top;
double ans=0;
double dis2(P a,P b){
	return (a.x-b.x)*(a.x-b.x)+(a.y-b.y)*(a.y-b.y);
}
double xmult(P a,P b,P o){
	return (a.x-o.x)*(b.y-o.y)-(a.y-o.y)*(b.x-o.x);
}
double cmp(P a,P b){
	return xmult(a,b,p[1])>0||xmult(a,b,p[1])==0&&dis2(a,p[1])<dis2(b,p[1]);
}
void graham(){
	sort(p+2,p+1+n,cmp);
	q[1]=p[1],q[2]=p[2];
	top=2;
	for(int i=3;i<=n;i++){
		while(xmult(q[top],p[i],q[top-1])<=0&&top>1)top--;
		q[++top]=p[i];
	}
}
void qiake(){
	q[top+1]=q[1];
	int now=2;
	for(int i=1;i<=top;i++){
		while(xmult(q[i+1],q[now],q[i])<xmult(q[i+1],q[now+1],q[i]))
		{
			now++;
			if(now>top)now=1;
		}
		ans=max(ans,dis2(q[now],q[i]));
	}
}
int main() {
	scanf("%d",&n);
	int t=1;
	for(int i=1;i<=n;i++){
```

```
            scanf("%lf%lf",&p[i].x,&p[i].y);
            if(p[t].y>p[i].y||p[t].y==p[i].y&&p[t].x>p[i].x)t=i;
        }
        swap(p[1],p[t]);
        graham();
        qiake();
        printf("%.6lf",sqrt(ans));
        return 0;
}
```