E5 D题 直直的多边形面积(计算几何)

题目描述

按照顺时针方向给出平面上的n个点（组成简单n边形），且简单n边形的每一条边均平行于坐标轴。求该n边形的面积，最多有5组数据。

题解思路1

首先尝试套计算多边形面积的板子，利用叉积算面积：

注意这里十分易错的就是当点的坐标都取最大值时，结果超出int的范围了。

代码1

```
#include<bits/stdc++.h>
#define ll long long
#define inf 1e100
#define eps 1e-10//用于浮点数正负判断，根据题目精度修改
#define rep(i,a,b) for(int i=a;i<=b;i++) //省略i=1 i<=n的for
using namespace std;
const int maxn=5e4+9;
const double pi = acos(-1.00);//圆周率
double sqr(double x){return x*x;}//求平方
int sgn(double x) {//判断浮点数正负
    if(fabs(x)<eps)return 0;
    if(x<0)return -1;
    return 1;
};

//使用Point时注意部分函数是返回新Point而非修改本身值
struct Point{
    double x,y;
    /*构造函数*/
    Point(){}
    Point(double xx,double yy){
        x=xx;y=yy;
    }
    /*重载点的基础运算符*/
    bool operator == (Point b)const{// 判断点相等
        return sgn(x-b.x) == 0 && sgn(y-b.y) == 0;
    }
    bool operator < (Point b)const{// 判断a和b的大小(先x后y)
        return sgn(x-b.x)== 0? sgn(y-b.y)<0:x<b.x;
    }
    Point operator -(const Point &b)const{// 返回xy分别做差后的新的点
        return Point(x-b.x,y-b.y);
    }
    Point operator +(const Point &b)const{// 返回xy分别求和后的新的点
        return Point(x+b.x,y+b.y);
    }
```

```
Point operator *(const double &k)const{// 返回xy分别数乘k后的新的点
    return Point(x*k,y*k);
}
Point operator /(const double &k)const{// 返回xy分别/k后的新的点
    return Point(x/k,y/k);
}
//叉积
double operator ^(const Point &b)const{// 注意叉乘的符号，返回数值
    return x*b.y - y*b.x;
}
//点积
double operator *(const Point &b)const{
    return x*b.x + y*b.y;
}
/*常用函数操作*/
double rad(Point a,Point b){/*当前点为p，求角apb大小*/
    Point p = *this;
    return fabs(atan2( fabs((a-p)^(b-p)),(a-p)*(b-p) ));
}
/*逆时针旋转90度*/
Point rotleft(){
    return Point(-y,x);
}
/*顺时针旋转90度*/
Point rotright(){
    return Point(y,-x);
}
//两点距离
double dis(Point p){
    return sqrt(sqr(x-p.x)+sqr(y-p.y));
}
double abs(){//距离原点的距离
    return sqrt(abs2());
}
double abs2(){//距离原点的距离平方
    return sqr(x)+sqr(y);
}
//改变向量长度变为r
Point trunc(double r){
    double l = abs();
    if(!sgn(l))return *this;
    r /= l;
    return Point(x*r,y*r);
}
//单位化
Point unit() { return *this/abs(); }
//IO
void input(){
    scanf("%lf%lf",&x,&y);
}
void output(){
    printf("%.7f %.7f\n",x,y);
}
//绕着p点逆时针旋转angle°
Point rotate(Point p,double angle){
```

```
        Point v = (*this) - p;
        double c = cos(angle), s = sin(angle);
        return Point(p.x + v.x*c - v.y*s,p.y + v.x*s + v.y*c);
    }
};
ll area(vector<Point> A, int n) {
    ll ans = 0;
    for (int i = 0; i < n; i++) ans += (A[i]^A[(i + 1) % n]);
    return ans/2;
}

int main()
{
    int t;
    scanf("%d",&t);
    while(t--){
        int n = 0;
        ll ans = 0;
        scanf("%d",&n);
        vector<Point> A(n+1);
        for(int i=n-1; i>=0; --i)
            scanf("%lf%lf",&A[i].x,&A[i].y);
        printf("%lld\n",area(A,n));
    }
    return 0;
}
```

题解思路2

这道题的数据较为简单，直接开数组表示横纵坐标也可以。按照边界点将图形分开成为多个三角形即可，这俩第n+1个点实质上就是第一个点。

注意：x,y为全局数组时，每新录入一组数据之前都要memset一下它们。

代码2

```
#include <bits/stdc++.h>
typedef long long ll;
using namespace std;
ll n;
ll x[400005], y[400005];
ll ans;
int main()
{
    int t;
    scanf("%d",&t);
    while(t--){
        memset(x,0,sizeof(x));
        memset(y,0,sizeof(y));
        n = 0; ans = 0;
        scanf("%d",&n);
        for(int i=1;i<=n;++i)
            scanf("%lld%lld",&x[i],&y[i]);
        x[n+1] = x[1], y[n+1] = y[1];
        for(int i=1;i<=n;++i)
```

```
            ans += (x[i]*y[i+1] - x[i+1]*y[i]);
        printf("%lld\n",abs(ans/2));
    }
    return 0;
}
```