卫星探测

【问题描述】

A 国最近检测到了 B 国内有不正常的辐射，经调查发现，这是因为 B 国正在耗资百亿研制新式武器——连环阵。可是，由于 B 国对此武器的高度保密措施，A 国的间谍甚至无法确定出连环阵的具体位置。不过，A 国的卫星还是可以找出连环阵所在的基地的。我们现在知道该基地是一个边上含有放射性物质的凸多边形。经研究发现，在这个凸多边形所在的平面内，它具有如下性质：

- 包含坐标原点；
- 任意两条边不在同一直线上；
- 没有和 *x* 轴或 *y* 轴平行的边；
- 所有顶点的 *x* 坐标和 *y* 坐标都是整数。

现在 A 国可以通过卫星发出无限大的扇形探测波，与该凸多边形所在平面交于一条直线。不过该直线不是与 *x* 轴平行，就是与 *y* 轴平行。通过反射信号，我们可以确定该直线与凸多边形的交点个数和所有交点的坐标（如果有的话）。

现在，需要你写一个程序，通过控制卫星发出的探测波来确定这个凸多边形。

【交互方法】

本题是一道交互式题目，*你的程序应当和测试库进行交互，而不得访问任何文件。*测试库提供 3 组函数，分别用于程序的初始化，与测试库的交互，以及返回结果。它们的用法与作用如下：

- `prog_initialize` **必须先调用**，但只能调用一次，用作初始化测试库；
- 测试库提供两个函数 `ask_x` 和 `ask_y` 作为与测试库交互的方式。其中 `ask_x(`x_0`, `y_1`, `y_2`)` 的的作用是询问直线 $x=x_0$ 和多边形的交点，`ask_y(y0, x1, x2)` 的作用是询问直线 $y=y_0$ 与多边形的交点，函数的返回值是交点的个数。`ask_x(`x_0`, `y_1`, `y_2`)` 调用后，y_1 和 y_2 被赋值为交点的 y 坐标；`ask_y(`y_0`, `x_1`, `x_2`)` 调用后，x_1 和 x_2 被赋值为交点的 x 坐标。如果只有一个交点，那么 x_1 和 x_2 或 y_1 和 y_2 的值相同；如果没有交点，那么 x_1 和 x_2 或 y_1 和 y_2 的值没有意义。
- 最后的一组函数是你的程序用来向测试库返回结果的。这里包括返回多边形面积的 `ret_area(s)`，返回多边形顶点数目的 `ret_n(n)`，返回多边形顶点坐标的 `ret_vertex(x, y)`。需要注意的事，这里 `ret_area` 是必须先于 `ret_n` 调用的，而 `ret_n` 又是必须先于 `ret_vertex` 调用的。不合适的调用方式将会强制你的程序非法退出。这里你需要在调用 `ret_n` 后调用 n 次 `ret_vertex` 返回多边形的顶点，但需要注意的是，如果你用 `ret_n` 返回的结果是错误的，那么测试库将会马上终止你的程序，并不接受下面的结果；同样的，如果 `ret_vertex` 中返回了错误的结果，那么测试库也会马上终止你的程序。如果 `ret_vertex` 的结果均是正确的，那么测试库将会在你返回最后一个顶点坐标后终止你的程序。
- **注意：你需要按照逆时针顺序返回所有顶点的坐标，不过你可以从任意一个顶点开始。**

【对使用 Pascal 选手的提示】

你的程序应当使用如下的语句引用测试库。

```
uses detect_lib;
```

测试库使用的函数原型为：

```
procedure prog_initialize;
function ask_x(const x0: longint; var y1, y2: double): longint;
function ask_y(const y0: longint; var x1, x2: double): longint;
procedure ret_area(const s: double);
procedure ret_n(const n: longint);
procedure ret_vertex(const x, y:longint);
```

【对使用 C/C++选手的提示】

你应当建立一个工程，把文件 libdetect.o 包含进来，然后在程序头加上一行：

```
#include “detect_lib.h”
```

测试库使用的函数原型为：

```
void prog_initialize();
int ask_x(int x0, double *y1, double *y2);
int ask_y(int y0, double *x1, double *x2);
void ret_area(double s);
void ret_n(int n);
void ret_vertex(int x, int y);
```

【数据说明】

如果凸多边形的坐标如右图所示，那么一种可能得满分的调用方案如下：

Pascal 选手的调用方法	C/C++选手的调用方法	说明
Prog_initialize;	prog_initialize();	初始化程序
ask_x(-6, y1, y2);	ask_x(-6, &y1, &y2);	返回值 1，$y_1=2$，$y_2=2$
ask_x(-5, y1, y2);	ask_x(-5, &y1, &y2);	返回值 2，$y_1=3.4$，$y_2=-5$
ask_y(2, x1, x2);	ask_y(2, &x1, &x2);	返回值 2，$x_1=-6$，$x_2=16$
ask_y(-20, x1, x2)	ask_y(-20, &x1, &x2)	返回值 0，x_1、x_2 中的值无意义
ret_area(241.5);	ret_area(241.5);	返回面积
ret_n(5);	ret_n(5);	返回 n
ret_vertex(8, -9);	ret_vertex(8, -9);	返回顶点(8, -9)
ret_vertex(16, 2);	ret_vertex(16, 2);	返回顶点(16, 2)
ret_vertex(-1, 9);	ret_vertex(-1, 9);	返回顶点(-1, 9)
ret_vertex(-6, 2);	ret_vertex(-6, 2);	返回顶点(-6, 2)
ret_vertex(-5, -5);	ret_vertex(-5, -5);	返回顶点(-5, -5)

注意，该例子只是对库函数的使用说明，并没有算法上的意义。

这里 n 最大为 200，x、y 坐标在[-10000, 10000]这个区间内。

【评分方法】

如果你的程序有下列情况之一，得 0 分：

- 访问了任何文件(包括临时文件)或者自行终止；
- 非法调用库函数；

- 让测试库异常退出。

否则每个测试点你的得分按这样来计算：包括顶点数提交正确的 1 分，面积提交正确的 2 分，顶点坐标完全正确的 2 分，分数累计。剩下的 5 分将根据你调用 `ask_x` 和 `ask_y` 的总次数进行评判，公式如下：

这里 *x* 为你的程序调用的 `ask_x` 和 `ask_y` 的次数，*score* 为你的得分。

【你如何测试自己的程序】

1. 在工作目录下建立一个文件叫做 detect.in，文件的第一行包括一个整数 *n* 为顶点的数目，以下 *n* 行每行两个整数**按照逆时针方向**给出凸多边形的顶点坐标；
2. 执行你的程序，此时测试库会产生输出文件 detect.log，该文件中包括了你程序和库交互的记录和最后的结果；
3. 如果程序正常结束，detect.log 的最后一行会给出你的程序的分数；
4. 如果程序非法退出，则我们不保证 detect.log 中的内容有意义。