102.三角形包含(Triangle containment)

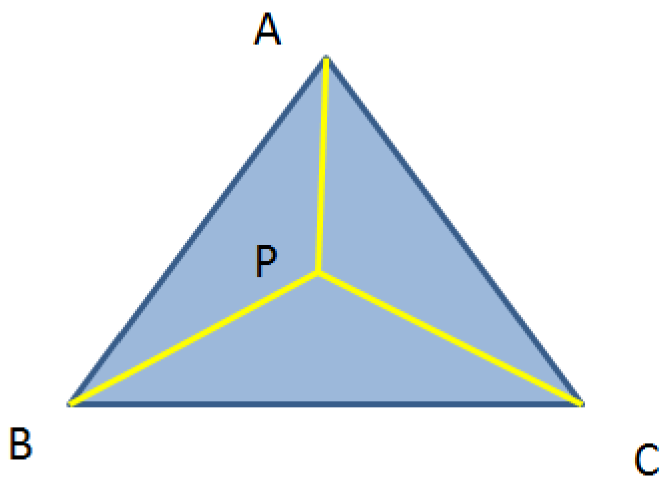
在笛卡尔坐标系中随机选取三个不同的点，其中 $-1000 \leqslant x,y \leqslant 1000$，然后用这三个点组成一个三角形。考虑下面两个三角形：

$$A(-340,495),B(-153,-910),C(835,-947)$$
$$X(-175,41),Y(-421,-714),Z(574,-645)$$

可以验证三角形 ABC 包含直角坐标系的原点，而三角形 XYZ 不包含。文本文件 triangles.txt 包含 1000 个随机三角形的顶点坐标，试问这一千个三角形中有多少个包含原点。

注：文件中的前两个三角形就是上面举的这两个例子。

分析：这是一道经典的计算机图形学入门问题，验证特定的点是否位于一个三角形中一般三种方法。第一种是面积法，即如果某个点在三角形内部，比如说点 P



在三角形 ABC 内部，则以点 P 为顶点的三个三角形 PAB、PAC 与 PBC 的面积之和等于三角形 ABC 的面积。如果点 P 在三角形外部，这个关系就不满足，所以

可以用这个性质来判断某个点是否在三角形内部。第二种和第三种方法分别是向量法与重心坐标法，相对来说，他们的效率要比面积法要高，但原理和计算也更为复杂一些。考虑到这道题的问题规模不大，面积法的效率已经让人很满意了，所以我使用了面积法。对后两种方法感兴趣的同学可以上网自行搜索。

使用面积法的核心问题是根据三角形的顶点坐标计算它的面积，可以直接套用相应的公式。设三角形 ABC 的顶点坐标分别为$A(x_1,y_1),B(x_2,y_2),C(x_3,y_3)$，则三形的面积为：

$$S_{\triangle ABC}=\frac{1}{2}|(x_1-x_3)(y_2-y_1)-(x_1-x_2)(y_3-y_1)|$$

根据这个公式，我们就可以用三角形的顶点坐标求出它的面积，然后再使用面积法判断原点是否位于三角形内部。代码如下：

```
# time cost = 9.9 ms ± 310 µs

def area_of_triangle(xa,ya,xb,yb,xc,yc):
    area = abs((xa-xc)*(yb-ya)-(xa-xb)*(yc-ya)) / 2
    return area

def main():
    counter = 0
    with open('ep102.txt') as f:
        coords = [x.strip().split(',') for x in f.readlines()]
        for coord in coords:
            xa,ya,xb,yb,xc,yc = [int(x) for x in coord]
            abc = area_of_triangle(xa,ya,xb,yb,xc,yc)
            aob = area_of_triangle(xa,ya,0,0,xb,yb)
            aoc = area_of_triangle(xa,ya,0,0,xc,yc)
            boc = area_of_triangle(xb,yb,0,0,xc,yc)
            if abc == aob + aoc + boc:
                counter += 1
    return counter
```