**计算机图形学 Homework3**

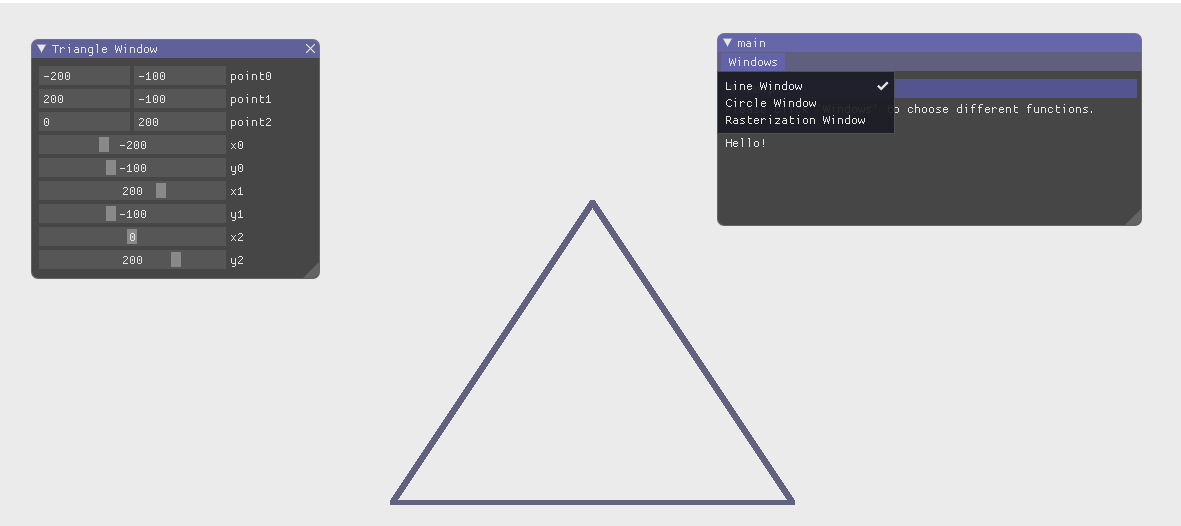
15331416 数字媒体技术方向 赵寒旭

**1. 运行结果**

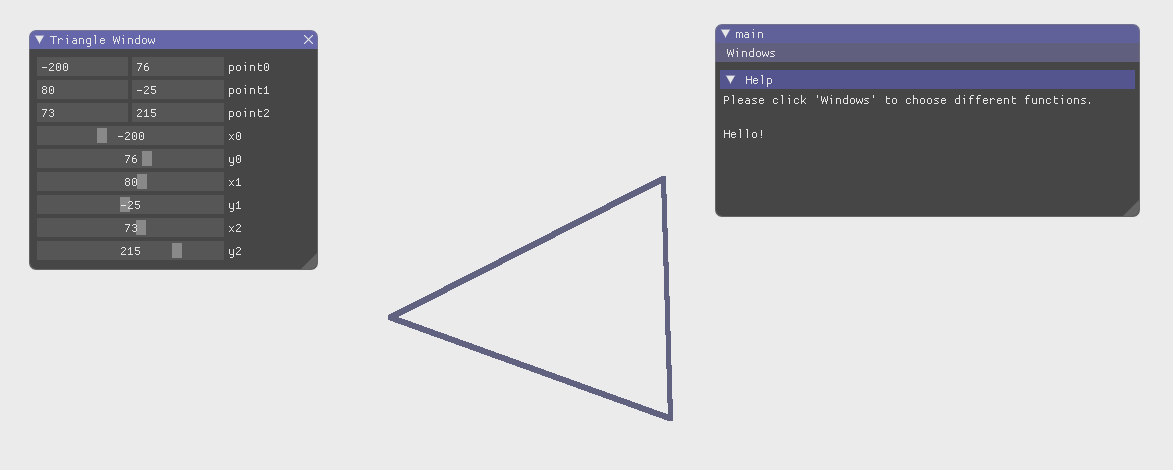
窗口大小初始化为1200\*800。

**1.1三角形边框绘制**

初始状态：point0 (-200, -100), point1 (200, -100), point2 (0, 200)

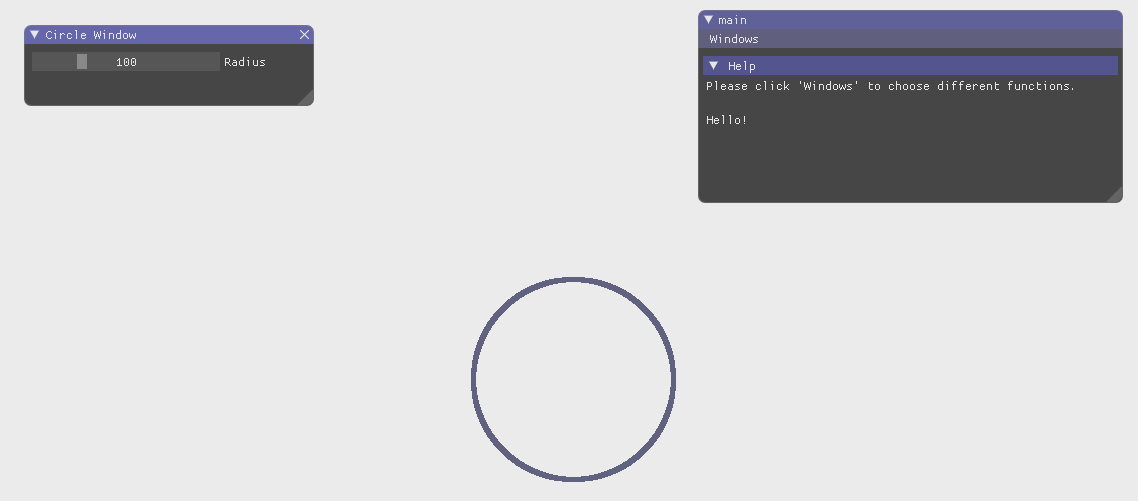


随意调整三角形顶点位置：



**1.2画圆及调整大小**

初始状态：圆心(0, 0) 半径100

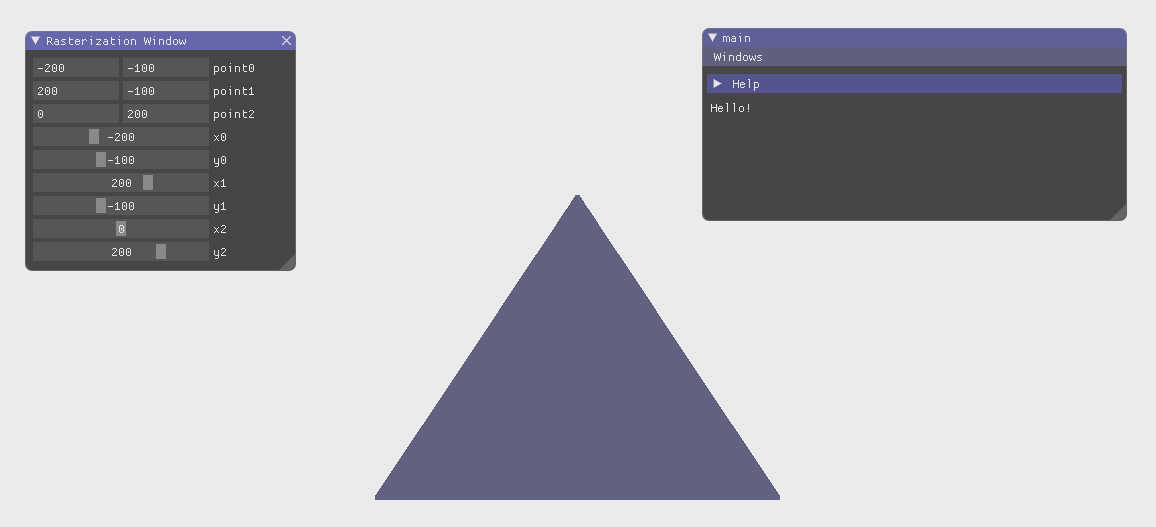


随意拉动滑块至 Radius=264

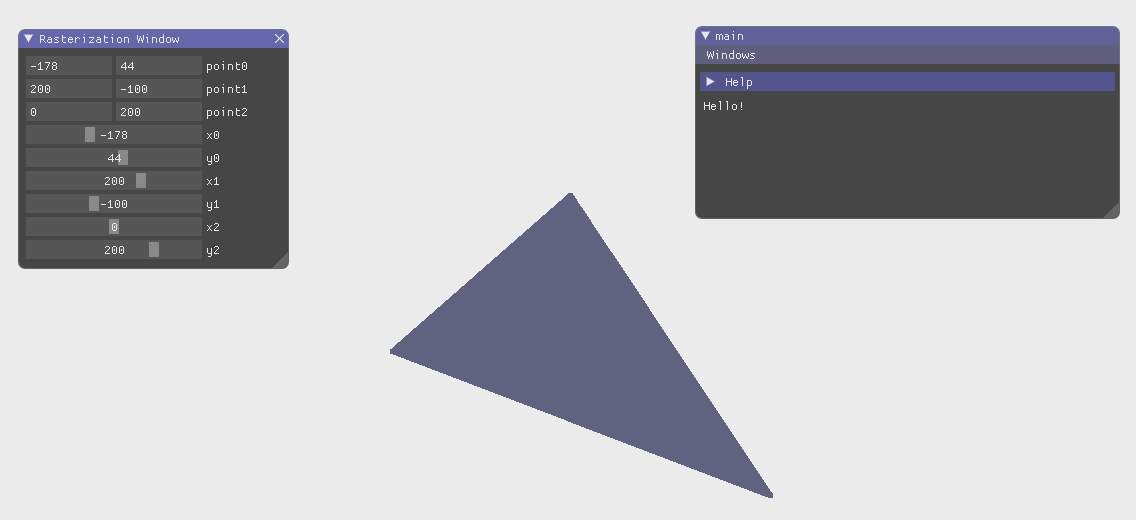


**1.3用区别于背景色的颜色填充三角形**

初始状态：point0 (-200, -100), point1 (200, -100), point2 (0, 200)



随意调整三角形顶点位置：



**2. 实现思路**

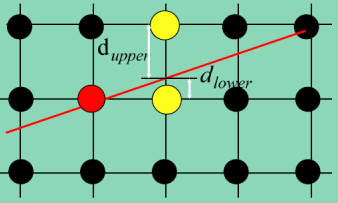
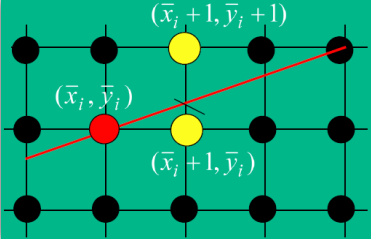
**2.1三角形边框的绘制**

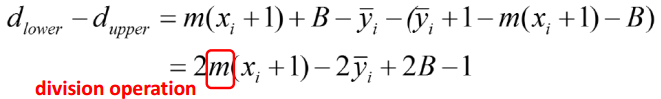
**2.1.1算法描述**

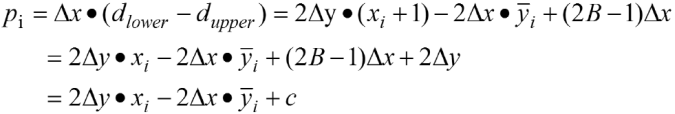
1）Bresenham直线算法

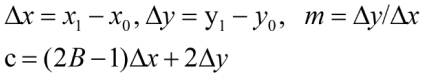
（1）一种典型的情况

输入起点终点的坐标A和B，我们的目标是求出两点连线上的所有像素点的坐标。首先考虑一种典型的情况：且：

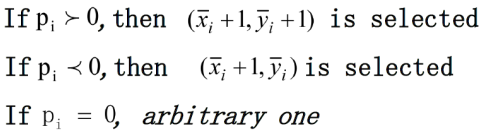
连线如下图：





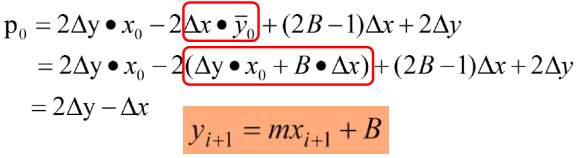


选择方法：根据距离上方和下方点的距离取最接近的一个

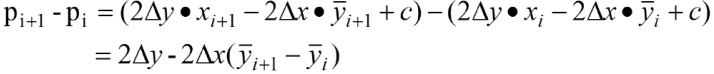


此时要计算仍然需要做多次数乘和求和运算，为了简化运算，我们可以使用迭代的方法根据之前的数据计算出当前的值。

首先计算初始值：



后项减前项可得：



可知迭代公式如下：

当，有，此时

当，有，此时

综上所述可得如下算法，用于获得连线上所有点的坐标：

① 存入起点的x，y坐标，计算，在末尾存入终点坐标

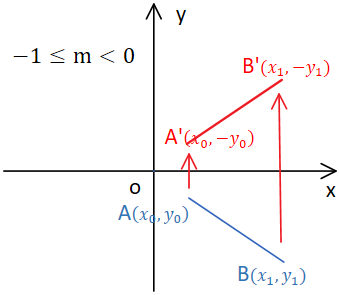
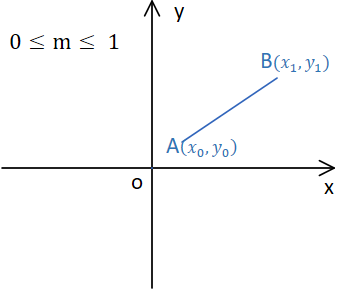
② 计算，，，，

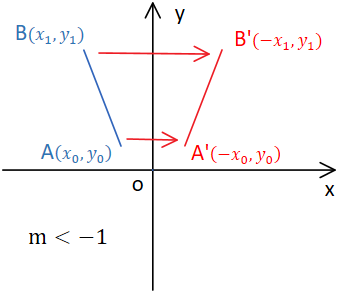
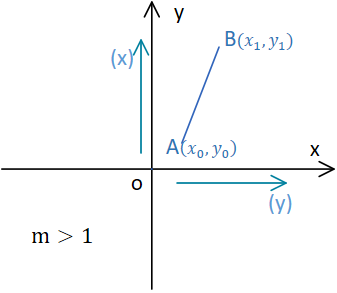
③ 若，存入，更新

④ 若，存入，更新

⑤ 重复③④，直到算至终点坐标前一点（时停止）

（2）其他情况的推广





时，先做关于x轴的镜像翻转，使得符合典型情况，在计算完一系列y坐标后再取反得到实际值。

与前两种情况不同，在时，y坐标递增，x坐标根据迭代关系逐步得到。此时观察图像可知，若把y轴看作典型情况中的x轴，起点A到终点B的连线符合典型情况。

特殊地，当斜率不存在时，我们可以不通过Bresenham算法直接递增得到y坐标。

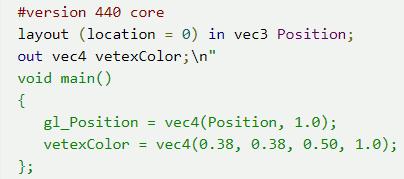
2）由顶点连线组成三角形边框

由1）所述，我们可以由两个端点生成线段上所有点的坐标，对三角形三个点两两求出连线上点的坐标即可构造出三角形。

**2.1.2代码实现**

1）着色器设置

因为本次任务不涉及颜色变化，故在顶点着色器中直接定义了颜色值。



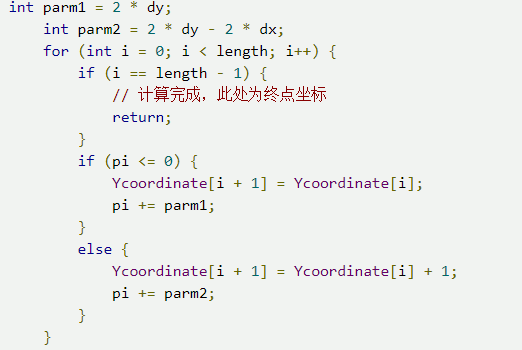
2）迭代坐标生成函数getYcoordinate



输入：待获得的迭代数组Ycoordinate，初始值pi，dx，dy，数组长度length

输出：迭代赋值完成的数组Ycoordinate

作用：根据Bresenham算法循环迭代得到坐标值



3）二维坐标生成函数BresenhamLine



输入：起点A和终点B的坐标，，，

输出：逐个存放线段上所有点x，y坐标值的数组p

作用：根据输入两点坐标得到线段上所有点的二维坐标

具体实现分5种情况分别处理，参见2.1.1里Bresenham直线算法描述中（2）其他情况的推广所描述的4种斜率取值范围及斜率不存在的情况。

4）归一化函数normalize



输入：存有全部x,y坐标值的数组p，窗口宽度width，窗口高度height

输出：归一化后的vector<float> q

作用：用于将生成的坐标数组中所有的值根据窗口大小归一到-1到1

由int转到float的方法：



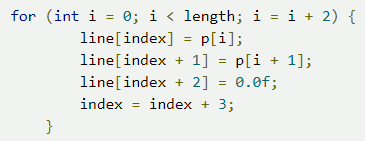
5）顶点数组生成函数getRealCoordinate



输入：由BresenhamLine并经normalize归一化得到的vector<float> p和数组长度lengtjh

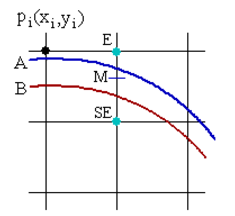
输出：含有每个点三维坐标的动态数组

作用：把含有x,y坐标的vector转换成float\*顶点数组(填充z值0.0f)



**2.2圆的绘制**

**2.2.1算法描述**

1）Bresenham算法绘制1/8圆

和画直线的算法相似，取右边点和右下点的中点M，判断点在圆内还是圆外，再判断选取E还是SE。

构造判别函数

有判别式：

① 若，取E为下一个点，此时下一个判别式：

将带入可得：

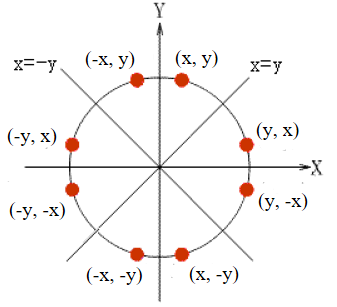
② 若，取SE为下一个点，此时下一个判别式：

将d带入可得：

特别地，在点，可以推出判别式的初始值：

为了避免浮点运算，可以将d的计算放大两倍，同时将初始值改为，乘二运算也可以用左移一位快速代替。

2）8分法生成整圆



只要获得圆1/8一段圆弧上点的坐标，就可以根据和圆心的相对位置得到整个圆的坐标。

**2.2.2代码实现**

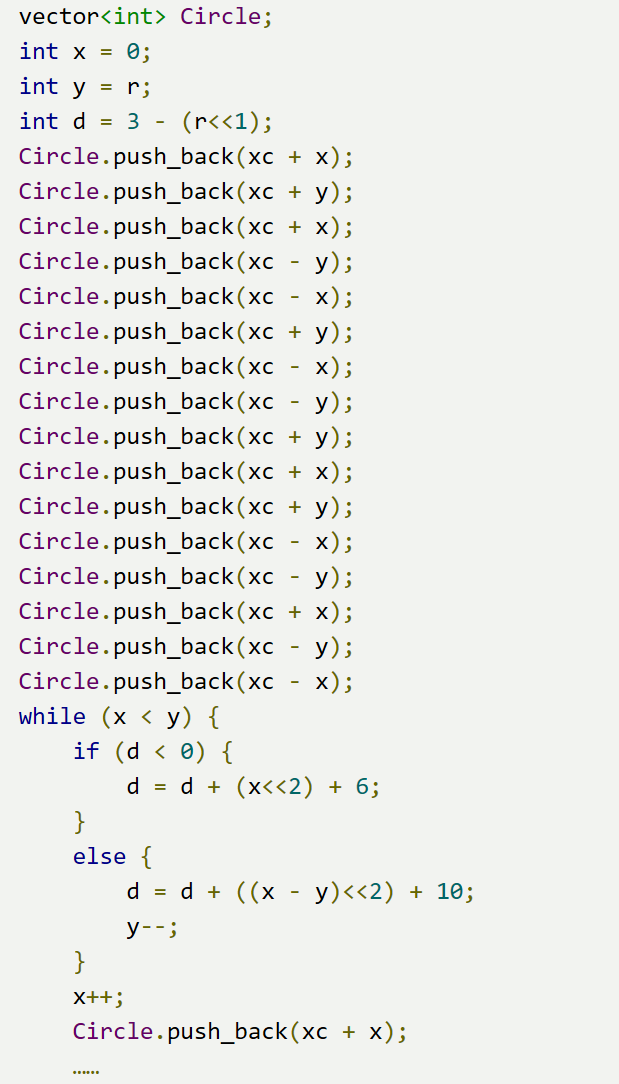
（1）圆二维坐标生成函数



输入：圆心坐标和半径r

输出：逐个存放圆弧上所有点x，y坐标值的数组Circle

作用：根据输入得到圆上所有点的二维坐标



循环中仅产生1/8的圆弧，其余都相对圆心自动生成。

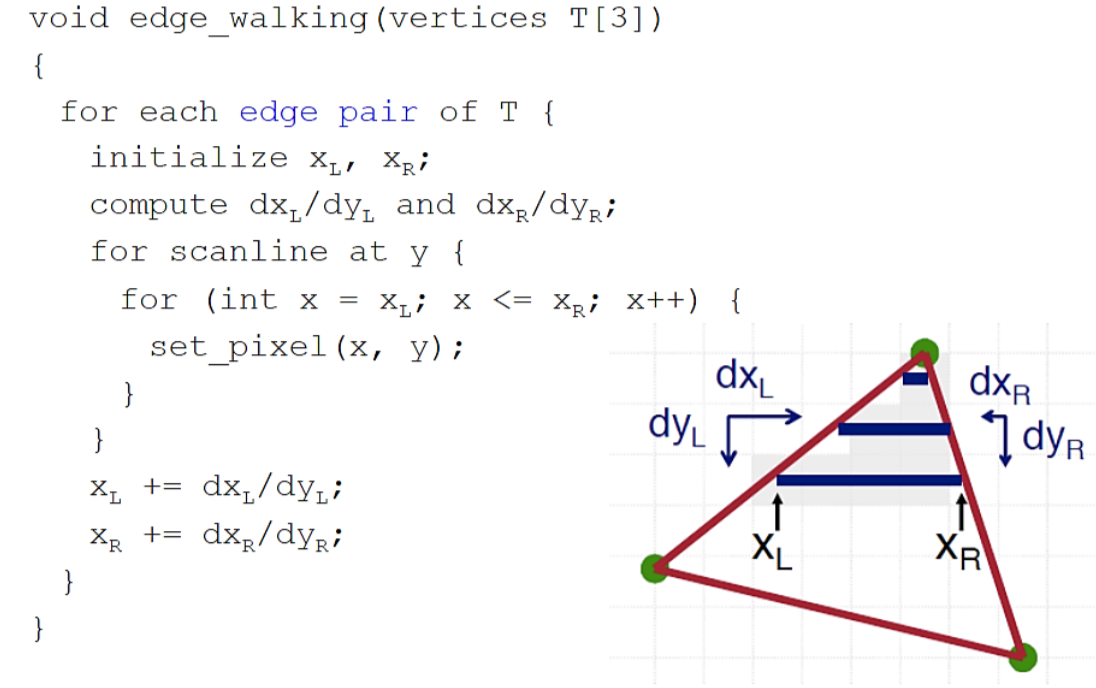
（2）归一化和顶点数组生成

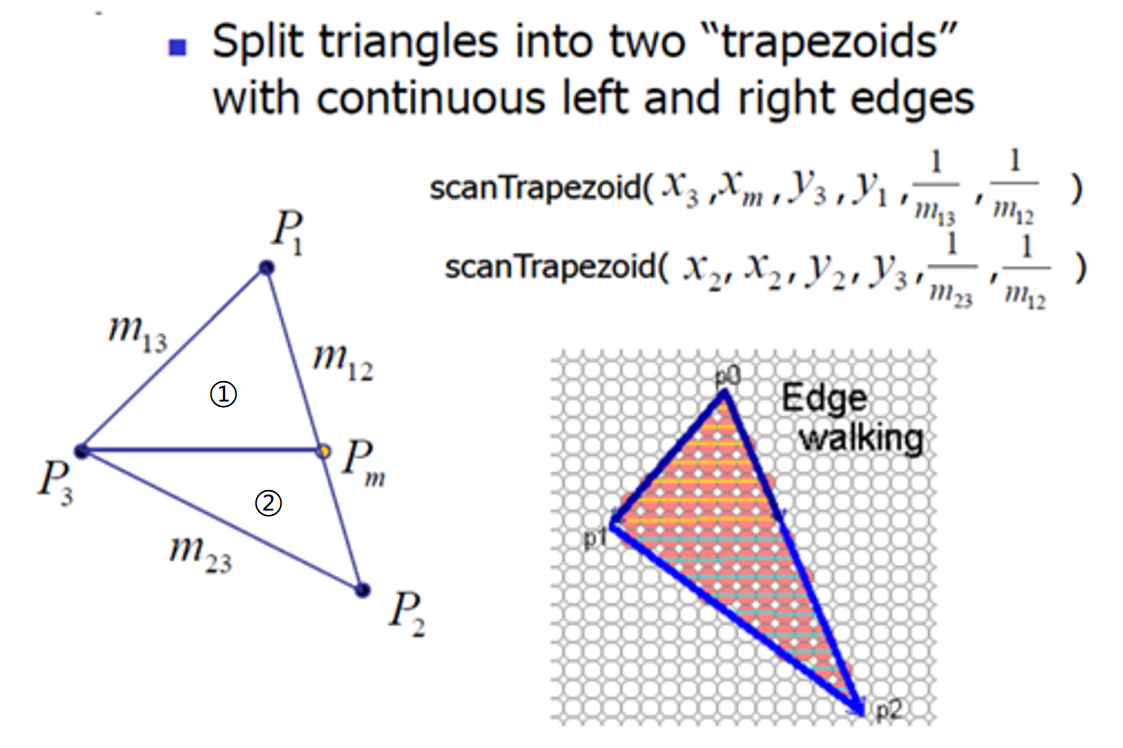
同2.1中方法。

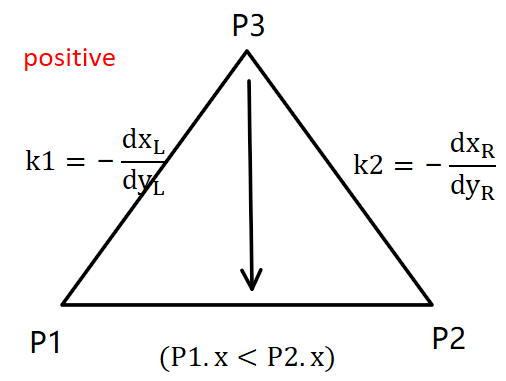
**2.3 三角形光栅转换算法**

**2.3.1算法描述**

参考ppt中提到的edge walking算法，根据三角形的形状特点分情况绘制：





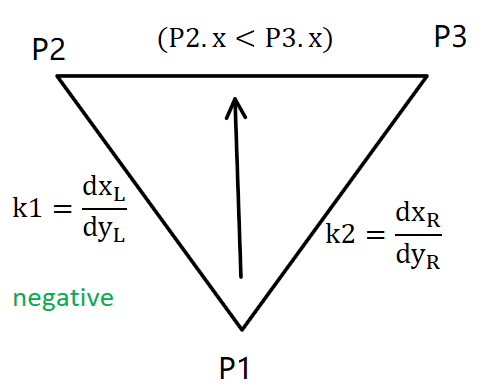
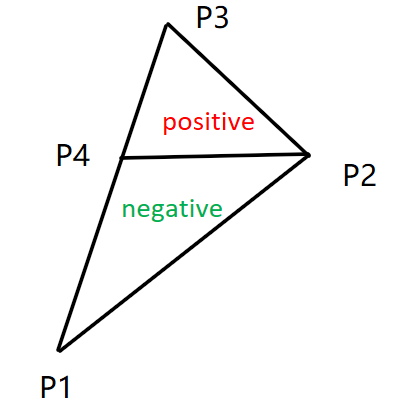
所有的三角形都可以看作是如上图①和②这两种形式三角形的组合，用positive形容如①形式的三角形，用negative来形容如②形式的三角形。

以绘制positive三角形为例：

将和初始化为P3.x，综合考虑客观坐标系和方向问题，取，的相反数为其赋值，我们可以得到迭代公式：

随着y值从P3.y到P1.y逐步减小，逐步减小，逐步增大。

每次将和中间的x坐标和对应的y坐标放入数组，最终得到三角形内部所有点的坐标。

绘制negative三角形的方法同样，此时从P1.y到P2.y，k1，k2不必取反。

绘制普通三角形时，可以看作两个特殊三角形的组合，先求出根据P1P3的直线

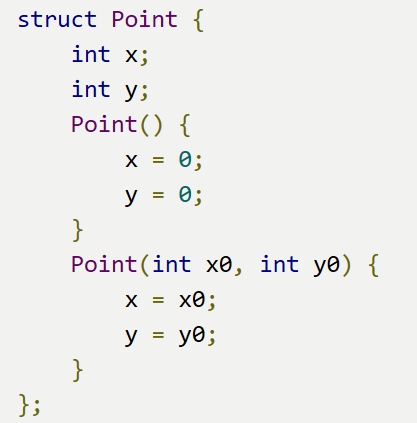
方程，计算y=P2.y与P1P3的交点，求出P4的坐标。

代入：

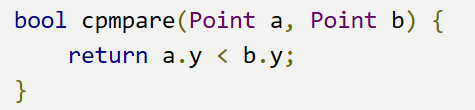
P4坐标：

**2.3.2 代码实现**

1) 结构体Point



x，y是Point对应的坐标值，函数compare使可以按照Point的数据成员y的大小用sort升序排序。



2）positive及negative三角形二维坐标生成函数

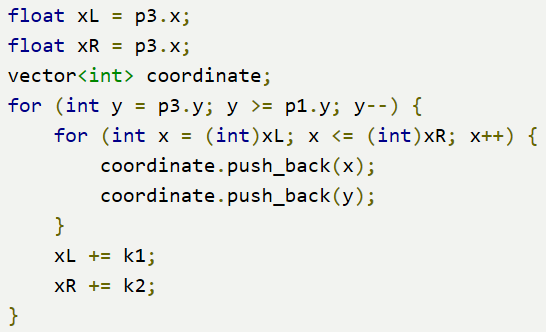
以positive为例：



输入：3个点

输出：逐个存放线段上所有点x，y坐标值的数组

作用：根据输入三点坐标得到positive三角形内部所有点的二维坐标



3）三角形二维坐标生成函数drawTriangle



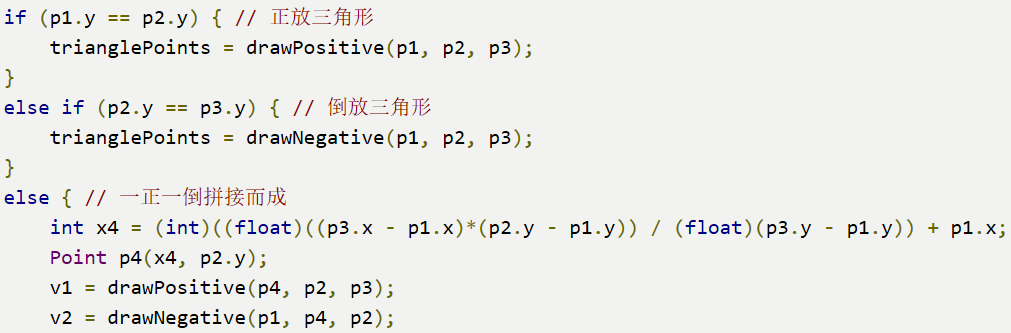
输入：3个点

输出：逐个存放线段上所有点x，y坐标值的数组

作用：根据输入三点坐标得到三角形内部所有点的二维坐标

首先按照y值升序排序：





分三种情况讨论，具体算法参考前文。