作业 2

一、作业描述

在上次作业中，虽然我们在屏幕上画出一个线框三角形，但这看起来并不是 那么的有趣。所以这一次我们继续推进一步——在屏幕上画出一个实心三角形， 换言之，栅格化一个三角形。上一次作业中，在视口变化之后，我们调用了函数 rasterize_wireframe(const Triangle& t)。但这一次，你需要自己填写并调用函数 rasterize_triangle(const Triangle& t)。

该函数的内部工作流程如下：

1. 创建三角形的 2 维 bounding box。

2. 遍历此 bounding box 内的所有像素（使用其整数索引）。然后，使用像素中心的屏幕空间坐标来检查中心点是否在三角形内。

3. 如果在内部，则将其位置处的插值深度值 (interpolated depth value) 与深度 缓冲区 (depth buffer) 中的相应值进行比较。

4. 如果当前点更靠近相机，请设置像素颜色并更新深度缓冲区 (depth buffer)。

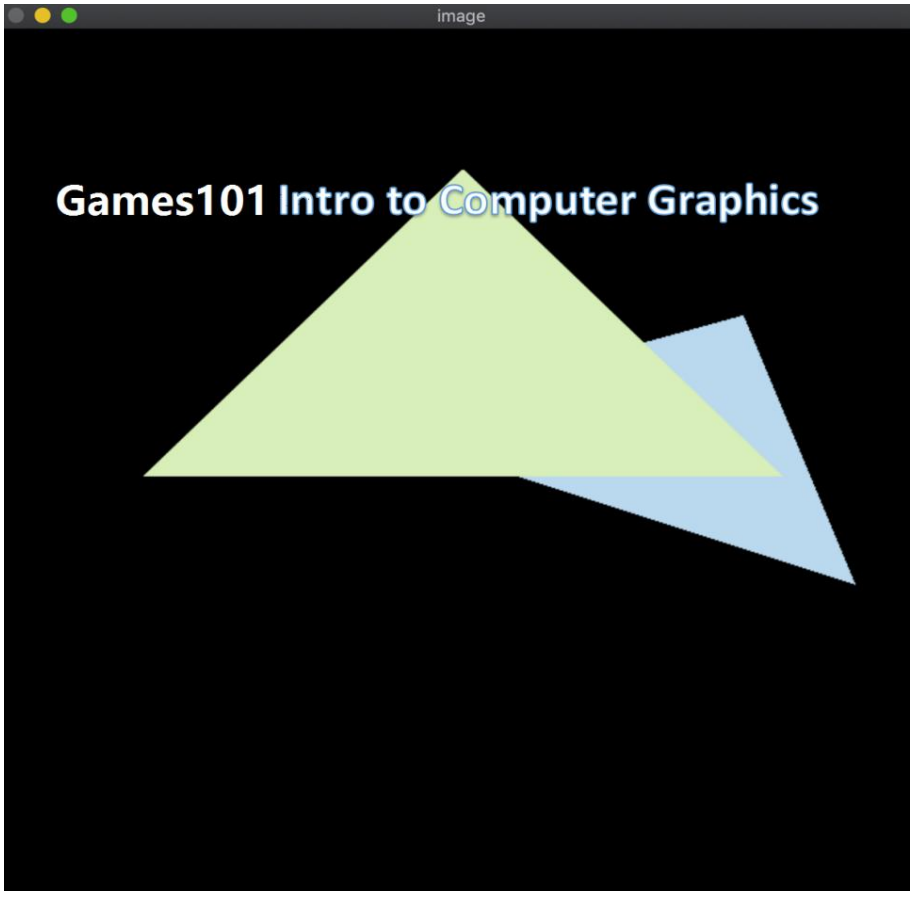
你需要修改的函数如下：

• rasterize_triangle()：执行三角形栅格化算法

• static bool insideTriangle()：测试点是否在三角形内。你可以修改此函数的定义，这意味着，你可以按照自己的方式更新返回类型或函数参数。

因为我们只知道三角形三个顶点处的深度值，所以对于三角形内部的像素，我们需要用插值的方法得到其深度值，我们已经为你处理好了这一部分，插值的深度值被储存在变量 z_interpolated 中。

请注意我们是如何初始化 depth buffer 和注意 z values 的符号。我们将 z 进行了反转，保证都是正数，并且越大表示离视点越远。在此次作业中，你无需处理旋转变换，只需为模型变换返回一个单位矩阵。最后，我们提供了两个 hard-coded 三角形来测试你的实现，如果程序实现正确，你将看到如下所示的输出图像：



在你自己的计算机或虚拟机上下载并使用我们更新的框架代码。你会注意到，在 main.cpp 下的 get_projection_matrix() 函数是空的。请复制粘贴你在第一次作业中的实现来填充该函数。

二、作业要求

1. 只需要提交源码文件(.h　.hpp　.cpp　.c 等文件格式)。不要将整个工程文件直接打包发来，只需要源代码，多交扣分，文件命名也请规范 ！
2. 请提交一份实验报告，模板见另一个文件，报告命名格式为：学号_姓名_实验报告 1。
3. 提交打包成一个压缩包，压缩包命名格式为：学号_姓名_作业 1（58119101_张三_作业 1）。
4. 只需要在 deadline 之前完成上述要求即可获得满分，额外实现不会加分。
5. 如有需求可以修改其他代码
6. 请在 deadline 前将压缩包发送至邮箱 tengyiqing@foxmail.com