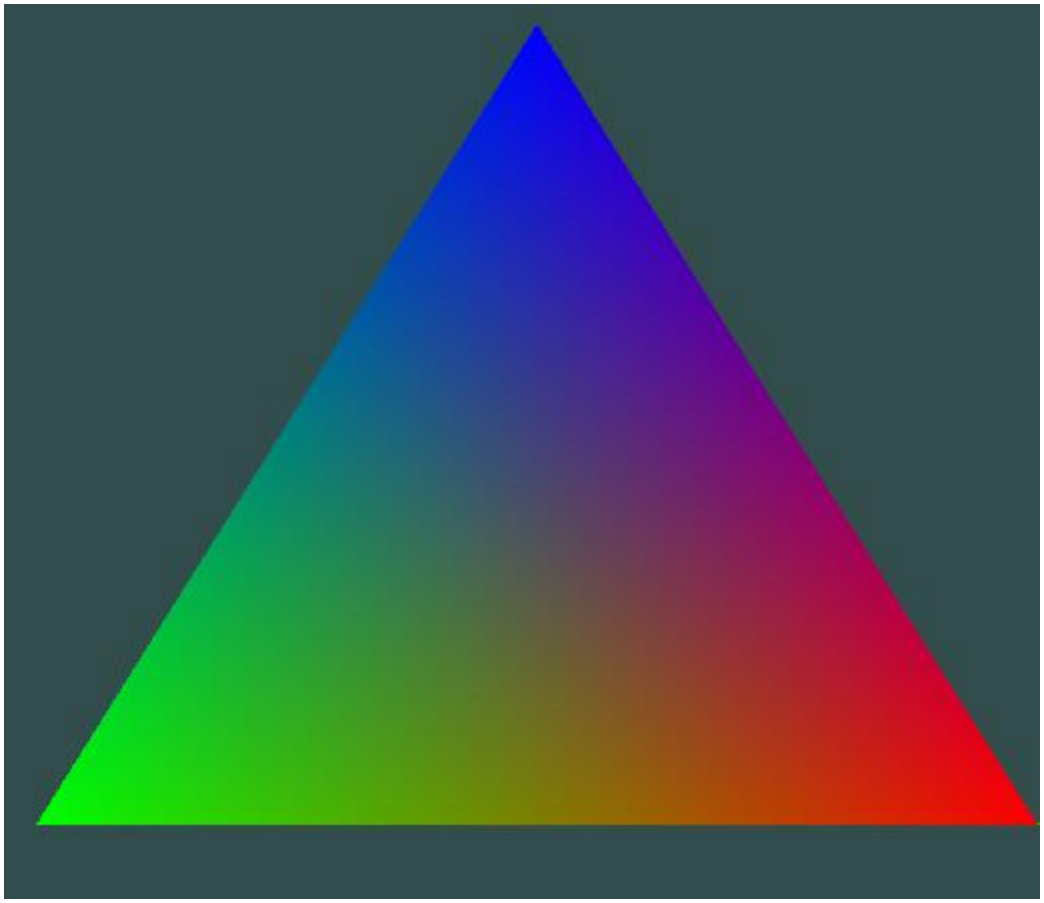
Homework 2

1. 使用 OpenGL(3.3 及以上)+GLFW 或 freeglut 画一个简单的三角形。
2. 对三角形的三个顶点分别改为红绿蓝，像下面这样。并解释为什么会出现这样的结果

第一题与第二题的运行结果如下图所示：



在代码中，我们只是对三角形的三个顶点分别渲染成红绿蓝，但最终的结果显示，除了三角形的三个顶点的红绿蓝色外，其它地方也呈现了不同的颜色。这是片段着色器中进行的所谓片段插值的结果。当渲染一个三角形时，光栅化阶段通常会造成比原指定顶点更多的片段。光栅会根据每个片段在三角形形状上所处相对位置决定这些片段的位置。

基于这些位置，它会插值所有片段着色器的输入变量。比如说，我们有一个线段，上面的端点是绿色的，下面的端点是蓝色的。如果一个片段着色器在线段的 70%的位置运行，它的颜色输入属性就会是一个绿色和蓝色的线性结合。更精确地说就是 30%蓝 + 70%绿。这正是在这个三角形中发生了什么。

在绘制三角形的过程中，有几个重要的函数意义需要理解。

以下是顶点缓冲对象的主要代码

```
unsigned int VBO;
glGenBuffers(1, &VBO);
glBindBuffer(GL_ARRAY_BUFFER, VBO);
glBufferData(GL_ARRAY_BUFFER, sizeof(vertices), vertices, GL_STATIC_DRAW);
```

我们使用 glGenBuffers 函数和一个缓冲 ID 生成了一个 VBO 对象，顶点缓冲对象的缓冲类型是 GL_ARRAY_BUFFER。这时，我们使用的任何在 GL_ARRAY_BUFFER 目标上的缓冲调用都会用来配置当前绑定的缓冲(VBO)。然后我们可以调用 glBufferData 函数，它会把之前定义的顶点数据复制到缓冲的内存中。

以下是索引缓冲对象的主要代码：

```
float vertices[] = {
0.5f, 0.5f, 0.0f, // 右上角
 0.5f, -0.5f, 0.0f, // 右下角
-0.5f, -0.5f, 0.0f, // 左下角
-0.5f, 0.5f, 0.0f // 左上角 };
unsigned int indices[] = { // 注意索引从 0 开始!
0, 1, 3, // 第一个三角形
1, 2, 3 // 第二个三角形
};
unsigned int EBO;
glGenBuffers(1, &EBO);
glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, EBO);
glBufferData(GL_ELEMENT_ARRAY_BUFFER, sizeof(indices), indices, GL_STATIC_DRAW);
glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, EBO);
glDrawElements(GL_TRIANGLES, 6, GL_UNSIGNED_INT, 0);
```

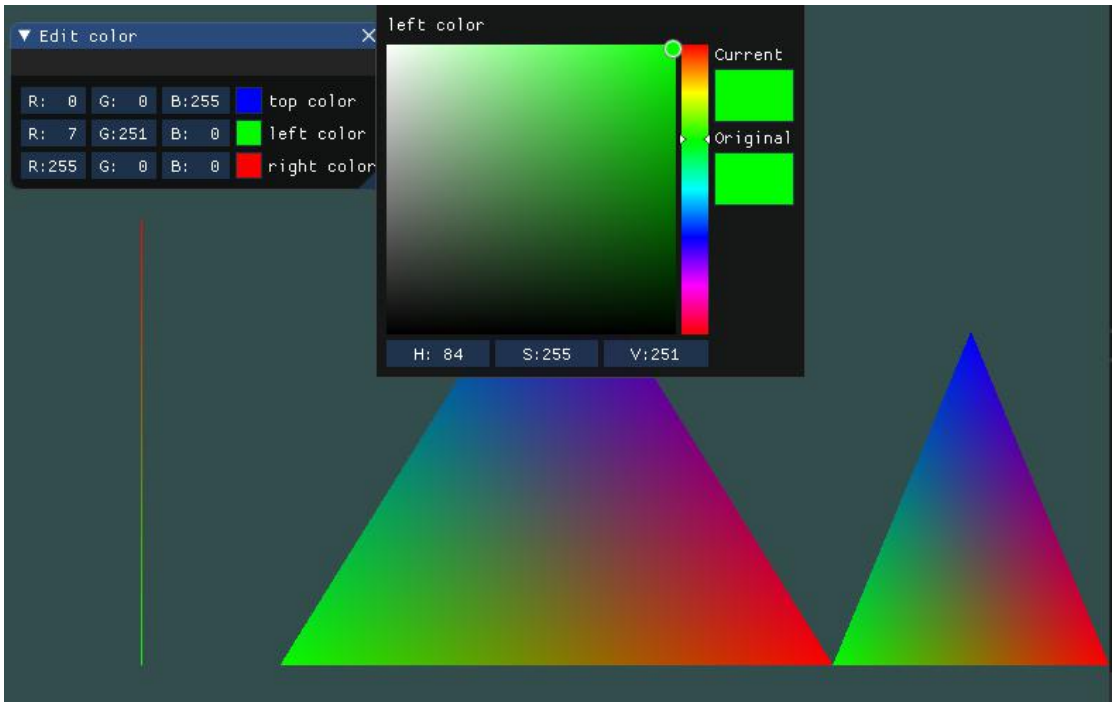
使用索引缓冲对象，可以方便地画多个三角形或其它图形。我们需要定义多个顶点，并且创建三角形顶点索引，然后创建 EBO 对象，然后绑定 EBO 然后用 glBufferData 把索引复

制到缓冲里，把缓冲的类型定义为 GL_ELEMENT_ARRAY_BUFFER。最后一件要做的事是用 glDrawElements 来替换 glDrawArrays 函数，来指明我们从索引缓冲渲染。

最后便可成功画得多个三角形。

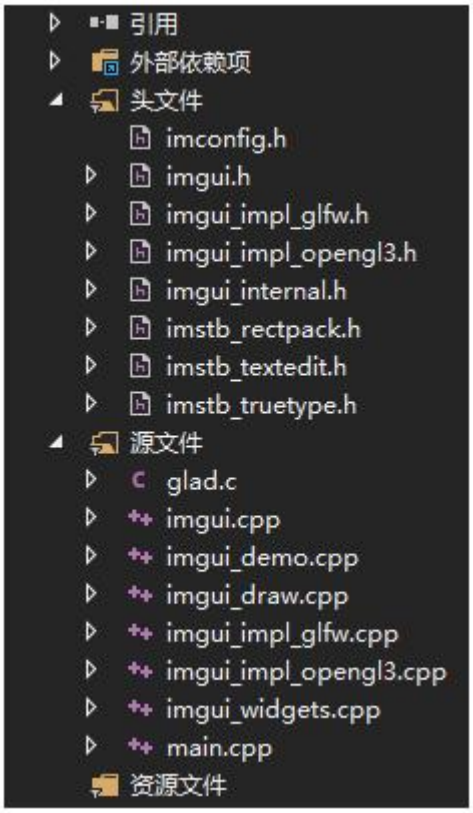
3. 给上述工作添加一个 GUI，里面有一个菜单栏，使得可以选择并改变三角形的颜色

增加了 GUI，根据 GUI 可以改变三角形三个角的颜色，如下图所示:



利用 ImGUI 实现颜色编辑器并不难。

首先要配置 ImGUI 环境，查看下载到本地的 ImGUI 库，添加下图中的库文件文件，使得项目结构变成下图：



然后修改 imgui_impl_opengl3.h 文件，按照前面 OpenGL 的官方教程，配置环境时，我们使用到了一个 glad 的库，而不是 gl3w 的库，所以在 imgui_impl_opengl3.h 文件中，把 `IMGUI_IMPL_OPENGL_LOADER_GL3W` 替换为 `IMGUI_IMPL_OPENGL_LOADER_GLAD`

最后增加颜色编辑器主要代码，如下所示：

```
//创建并绑定
ImGui const char* glsl_version = "#version 130";
IMGUI_CHECKVERSION(); ImGui::CreateContext();
ImGuiIO& io = ImGui::GetIO(); (void)io;
ImGui::StyleColorsDark();
ImGui_ImplGlfw_InitForOpenGL(window, true);
ImGui_ImplOpenGL3_Init(glsl_version);

//以下部分放在 while 循环中
imgui ImGui_ImplOpenGL3_NewFrame();
ImGui_ImplGlfw_NewFrame();
ImGui::NewFrame();
ImGui::Begin("Edit color", &show_window, ImGuiWindowFlags_MenuBar);
ImGui::ColorEdit3("top color", (float*)&topColor);
ImGui::ColorEdit3("left color", (float*)&leftColor);
ImGui::ColorEdit3("right color", (float*)&rightColor);
ImGui::End(); ImGui::Render();
int display_w, display_h;
```

```
glfwMakeContextCurrent(window);
glfwGetFramebufferSize(window, &display_w, &display_h);
glViewport(0, 0, display_w, display_h);
ImGui_ImplOpenGL3_RenderDrawData(ImGui::GetDrawData());
```

将 imgui 渲染窗口的代码放进 opengl 渲染三角形的 while 循环中，这样的话，当用鼠标选择颜色时，获取的颜色的数据，存在 topColor、leftColor、rightColor 中，然后就可以对三角形重新进行颜色渲染，以达到目的。

4. 加分项

加分项是多画三角形和线段，如下图所示：