# BASIC：

**动态效果见附视频文件 hw4.mp4。**

1. 画一个立方体(cube)：边长为 4，中心位置为(0, 0, 0)。分别启动和关闭深度测试 glEnable(GL_DEPTH_TEST) 、 glDisable(GL_DEPTH_TEST) ，查看区别，并分析原因。
   当启动深度测试的时候，效果表现上立方体是不透明的，即使设置了透明度，当关闭测试的，才能显示透明的效果。
   原因分析：启用了深度测试之后，OpenGL 只会绘制最前面的一层像素，当检查到将要描绘的像素没有被深度更高的像素遮挡时才会绘制。而当关闭时会继续绘制，从而能呈现透明的效果。

2. 平移(Translation)：使画好的 cube 沿着水平或垂直方向来回移动。
   效果见附动态图

3. 旋转(Rotation)：使画好的 cube 沿着 XoZ 平面的 x=z 轴持续旋转。
   效果见附动态图

4. 放缩(Scaling)：使画好的 cube 持续放大缩小。
   效果见附动态图

5. 在 GUI 里添加菜单栏，可以选择各种变换。
   效果见附动态图

6. 结合 Shader 谈谈对渲染管线的理解
   渲染管线是指渲染时流水线的工作流程，包括顶点处理、几何处理、裁剪、光栅化和片元操作等。这些工作都是在显卡内处理的，而 shader 就是使用现代的 GPU 编程技术来灵活地设计渲染管线的工作过程，使得开发人员可直接控制 GPU 的行为，更高效更灵活地实现更丰富的显示效果。

# Bonus

1. 将以上三种变换相结合，打开你们的脑洞，实现有创意的动画。比如：地球绕太阳转等。
   效果见附动态图

# 代码思路解析

附代码文件（本次作业的核心代码在 hw4.cpp 内）

1. 立方体画法
   使用八个顶点坐标，配合一个 12 个三角形的索引数组 EBO，绘制出立方体。
2. 平移、旋转、缩放变换的实现
   利用变换矩阵的方法

```
if (translate)
{
    trans = glm::translate(trans, glm::vec3(sin(time) / 2, 0, 0));
}
if (rotate)
{
    trans = glm::rotate(trans, (float)glfwGetTime(), glm::vec3(1.0f, 0.0f, 1.0f));
}
if (scale)
{
    trans = glm::scale(trans, glm::vec3(abs(sin(time)), abs(sin(time)), abs(sin(time))));
}
```