Hw-2: Solar System

姓名：刘家豪 学号：3160104463

为了更好的理解OpenGL中坐标系的变换，以及视角的转换。老师让我们实现简易太阳系。鉴于太阳系中行星与卫星的运行方式的相似性， 故在设计过程中，只需实现地球，月亮和太阳三个星体的运行即可。

1.1 初步思路

对于太阳系的实现，我们大致需要实现以下几个方面：

- 实现星体的绘制，即球体的绘制通过查询资料知晓，在OpenGL中可以通过函数`glutWireSphere()`进行线体球的绘制，通过函数`glutSolidSphere()`进行实体球的绘制。
- 星体的自转与公转，对于球体的旋转，(转速，转向，旋转平面)，通过前两次实验，我们知道，可以采用`glTranslatef()`进行平移，采用`glRotatef()`进行旋转。
- 视角的变换，通过课上学习，我们可以采用透视投影，以及观察点的变换来解决。
- 轨道的绘制，OpenGL库函数中没有圆的绘制函数，需要根据相应坐标系，自己设计一个圆的绘制函数，来绘制轨道。

通过查询相应资料，大致熟悉了上述操作相应函数的使用方法，并掌握了坐标系与视角的切换方式，便开始太阳系的绘制。

1.2 问题解决过程

1.2.1 球体的绘制

在这一阶段，单纯的绘制三个球体，太阳，地球，月亮。由于对材质，光照的不熟悉，故采用`glutWireSpere()`函数绘制线体球，效果较好。

通过调整相应的半径，经纬线数，成功完成这一步。

1.2.2 球体的自公转

为了实现球体的自转与公转，我们调用平移与旋转函数来实现。

```
glPushMatrix();
glRotatef((GLfloat)time/50, 0.0, 0.0, 1.0);
glColor3f(1.0, 0.0, 0.0);
glutWireSphere(1.0, 20, 20);
glBegin(GL_LINE_LOOP);
glColor3f(0.6, 0.5, 0.4);
glVertex3d(0.0, 0.0, 2.0);
glVertex3d(0.0, 0.0, -2.0);
glEnd();
drawCircle(2.121);
glPopMatrix();//draw sun
```

在这一过程中发现，当进行旋转的时候，视角容易出问题，于是我们进行`viewport`调整，使用投影变换，并对窗口的拉伸做处理。实现可以对球体自公转的观察。

```
glViewport(0, 0, w, h);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluPerspective(60.0, (GLfloat)w / (GLfloat)h, 1.0, 20.0);

if (w <= h)
    glOrtho(-1.0, 1.5, -1.5*(GLfloat)h / (GLfloat)w, 1.5*(GLfloat)h /
(GLfloat)w, -1.0, 1.0);
else
    glOrtho(1.5*(GLfloat)w / (GLfloat)h, 1.5*(GLfloat)w / (GLfloat)h, -1.0,
1.5, -1.0, 1.0);

glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt(5.0, 0.0, 2.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0);
```

1.2.3 轨道的绘制

由于观察的时候，头朝矢量(0, 0, 1), 故设计圆的绘制函数，如下：

```
void drawCircle(float radius)
{
    glBegin(GL_LINE_LOOP);
    for (int i = 0; i < point; i++)
    {
        glVertex3f(radius * cos(2 * pi / point * i), radius * sin(2 * pi /
point * i), 0.0);
    }
    glEnd();
}
```

通过调用绘制圆的函数，我们实现对星体轨道的模拟。

1.3 结果

通过利用课堂上所学知识，与查阅的响应资料，成功模拟太阳系中部分星体的运转，这一过程加深了自己对OpenGL中视角变换，坐标系变换的理解，同时进一步锻炼了自己利用OpenGL编程的能力。

具体实验结果，请老师运行工程，麻烦了。