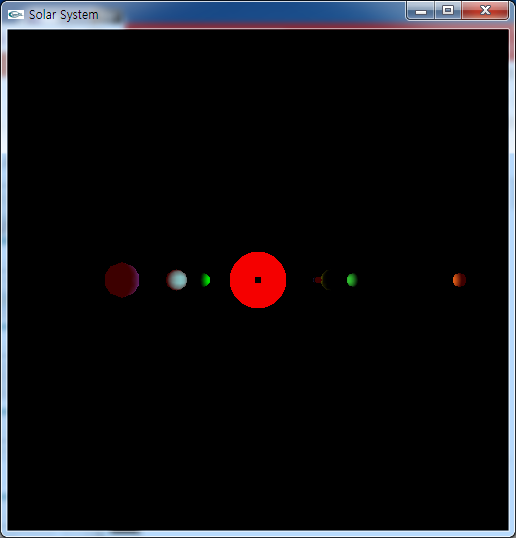
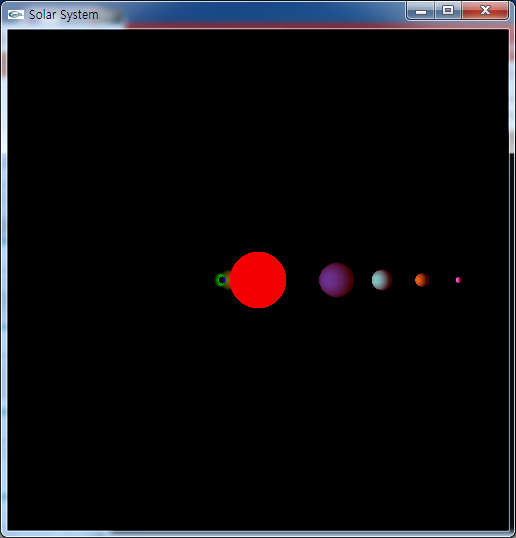
**HW4.**

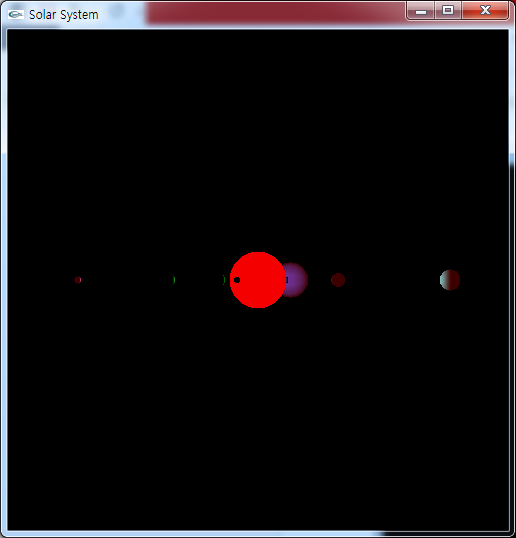
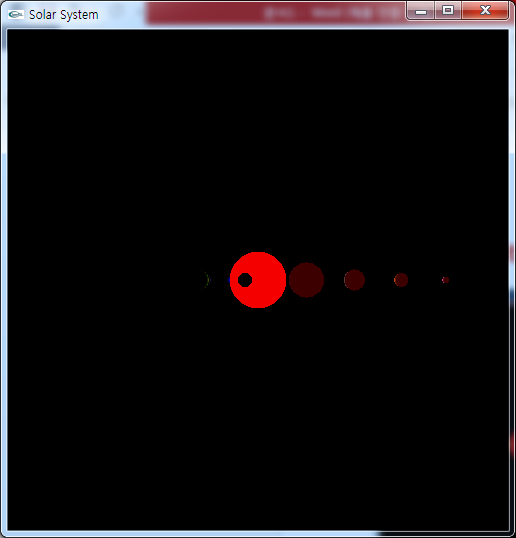
**Add the Effect of Sun Light to the Solar System**

**201133216**

**정유석**

**ScreenShot**





**Code**

#include <stdlib.h>

#include <time.h>

#include <gl/glut.h>

static int delay = 150; //Delay time

static int Su = 0, Geom = 0, Ji\_Day = 0,Ji\_Time=0, Hwa = 0, Mok = 0, To = 0, Cheon = 0, Hae = 0;

//use setting for planet

typedef struct {

GLfloat amb[4]; // ambient

GLfloat dif[4]; // diffuse

GLfloat spe[4]; // specular

GLfloat shi; // shininess

} Material;

//use setting for light

typedef struct {

GLfloat pos[4]; // position

GLfloat amb[4]; // ambient (self light)

GLfloat dif[4]; // diffuse (object color)

GLfloat spe[4]; // specular (spotlight)

} Light;

//Set light

Light light = {

{ 0.0, 0.0, 0.0, 1.0 }, // position

{ 1.0, 1.0, 1.0, 1.0 }, // ambient

{ 1.0, 1.0, 1.0, 1.0 }, // diffuse

{ 1.0, 1.0, 1.0, 1.0 }, // specular

};

//Set each planet light source

Material mat0 = {

{ 0.8, 0.0, 0.0, 1.0 }, // ambient

{ 1.0, 0.0, 0.0, 1.0 }, // diffuse

{ 1.0, 0.0, 0.0, 1.0 }, // specular

{ 100.0 }, // shininess

};

Material mat1 = {

{ 0.0, 0.0, 0.0, 1.0 }, // ambient

{ 0.0, 0.0, 1.0, 1.0 }, // diffuse

{ 0.0, 0.0, 1.0, 1.0 }, // specular

{ 100.0 }, // shininess

};

Material mat2 = {

{ 0.0, 0.0, 0.0, 1.0 }, // ambient

{0.0, 1.0, 0.0, 1.0 }, // diffuse

{ 0.0, 1.0, 0.0, 1.0 }, // specular

{ 100.0 }, // shininess

};

Material mat3 = {

{ 0.0, 0.0, 0.0, 1.0 }, // ambient

{ 0.4, 0.4, 0.0, 1.0 }, // diffuse

{ 0.4, 0.4, 0.0, 1.0 }, // specular

{ 100.0 }, // shininess

};

Material mat4 = {

{ 0.0, 0.0, 0.0, 1.0 }, // ambient

{ 0.2, 0.2, 0.5, 1.0 }, // diffuse

{ 0.2, 0.2, 0.5, 1.0 }, // specular

{ 100.0 }, // shininess

};

Material mat5 = {

{ 0.0, 0.0, 0.0, 1.0 }, // ambient

{ 0.2, 0.8, 0.2, 1.0 }, // diffuse

{ 0.2, 0.8, 0.2, 1.0 }, // specular

{ 100.0 }, // shininess

};

Material mat6 = {

{ 0.2, 0.0, 0.0, 1.0 }, // ambient

{ 0.2, 0.2, 0.6, 1.0 }, // diffuse

{ 0.2, 0.2, 0.6, 1.0 }, // specular

{ 100.0 }, // shininess

};

Material mat7 = {

{ 0.2, 0.0, 0.0, 1.0 }, // ambient

{ 0.3, 0.8, 0.8, 1.0 }, // diffuse

{ 0.3, 0.8, 0.8, 1.0 }, // specular

{ 100.0 }, // shininess

};

Material mat8 = {

{ 0.2, 0.0, 0.0, 1.0 }, // ambient

{ 0.7, 0.4, 0.1, 1.0 }, // diffuse

{ 0.7, 0.4, 0.1, 1.0 }, // specular

{ 100.0 }, // shininess

};

Material mat9 = {

{ 0.2, 0.0, 0.0, 1.0 }, // ambient

{ 0.9, 0.4, 0.9, 1.0 }, // diffuse

{ 0.9, 0.4, 0.9, 1.0 }, // specular

{ 100.0 }, // shininess

};

void MyDisplay() {

glOrtho(-0.7, 0.7, -0.7, 0.7, -0.7, 0.7); //Set size of the 3-dimensional

glEnable ( GL\_DEPTH\_TEST );

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT); //Clear color buffer

glMatrixMode(GL\_MODELVIEW); //Set modelview

glLoadIdentity(); //Set C (It use calculation of transformation)

//Draw the Sun

glMaterialfv( GL\_FRONT, GL\_AMBIENT, mat0.amb );

glMaterialfv( GL\_FRONT, GL\_DIFFUSE, mat0.dif );

glMaterialfv( GL\_FRONT, GL\_SPECULAR, mat0.spe );

glMaterialf ( GL\_FRONT, GL\_SHININESS, mat0.shi );

glutSolidSphere(0.08, 20, 16);

//Draw the Mercury

glRotatef((GLfloat)Su, 0.0, 1.0, 0.0); //Rotation to itself

glTranslatef(0.1, 0.0, 0.0); //Transition to 0.1

glMaterialfv( GL\_FRONT, GL\_AMBIENT, mat1.amb );

glMaterialfv( GL\_FRONT, GL\_DIFFUSE, mat1.dif );

glMaterialfv( GL\_FRONT, GL\_SPECULAR, mat1.spe );

glMaterialf ( GL\_FRONT, GL\_SHININESS, mat1.shi );

glutSolidSphere(0.01, 10, 8); //Draw sphere

//Draw the Venus

glLoadIdentity(); //Set the first position(sun)

glRotatef((GLfloat)Geom, 0.0, 1.0, 0.0);

glTranslatef(0.17, 0.0, 0.0);

glMaterialfv( GL\_FRONT, GL\_AMBIENT, mat2.amb );

glMaterialfv( GL\_FRONT, GL\_DIFFUSE, mat2.dif );

glMaterialfv( GL\_FRONT, GL\_SPECULAR, mat2.spe );

glMaterialf ( GL\_FRONT, GL\_SHININESS, mat2.shi );

glutSolidSphere(0.02, 10, 8);

//Draw the Earth

glLoadIdentity(); //Set the first position(above same)

glRotatef((GLfloat)Ji\_Day, 0.0, 1.0, 0.0);

glTranslatef(0.25, 0.0, 0.0);

glRotatef((GLfloat)Ji\_Time, 0.0, 1.0, 0.0);

glMaterialfv( GL\_FRONT, GL\_AMBIENT, mat3.amb );

glMaterialfv( GL\_FRONT, GL\_DIFFUSE, mat3.dif );

glMaterialfv( GL\_FRONT, GL\_SPECULAR, mat3.spe );

glMaterialf ( GL\_FRONT, GL\_SHININESS, mat3.shi );

glutSolidSphere(0.03, 10, 8);

glPushMatrix(); //It push now position because Represent a revolution of the moon around the earth.

//Draw the moon

glRotatef((GLfloat)Ji\_Time, 0.0, 1.0, 0.0);

glTranslatef(0.05, 0.0, 0.0);

glMaterialfv( GL\_FRONT, GL\_AMBIENT, mat4.amb );

glMaterialfv( GL\_FRONT, GL\_DIFFUSE, mat4.dif );

glMaterialfv( GL\_FRONT, GL\_SPECULAR, mat4.spe );

glMaterialf ( GL\_FRONT, GL\_SHININESS, mat4.shi );

glutSolidSphere(0.01, 10, 8);

glPopMatrix(); //because it set the first(Sun) position

glLoadIdentity();

//Draw the Mars

glRotatef((GLfloat)Hwa, 0.0, 1.0, 0.0);

glTranslatef(0.35, 0.0, 0.0);

glMaterialfv( GL\_FRONT, GL\_AMBIENT, mat5.amb );

glMaterialfv( GL\_FRONT, GL\_DIFFUSE, mat5.dif );

glMaterialfv( GL\_FRONT, GL\_SPECULAR, mat5.spe );

glMaterialf ( GL\_FRONT, GL\_SHININESS, mat5.shi );

glutSolidSphere(0.02, 10, 8);

//Draw the Jupiter

glLoadIdentity();

glRotatef((GLfloat)Mok, 0.0, 1.0, 0.0);

glTranslatef(0.44, 0.0, 0.0);

glMaterialfv( GL\_FRONT, GL\_AMBIENT, mat6.amb );

glMaterialfv( GL\_FRONT, GL\_DIFFUSE, mat6.dif );

glMaterialfv( GL\_FRONT, GL\_SPECULAR, mat6.spe );

glMaterialf ( GL\_FRONT, GL\_SHININESS, mat6.shi );

glutSolidSphere(0.05, 10, 8);

//Draw the Saturn

glLoadIdentity();

glRotatef((GLfloat)To, 0.0, 1.0, 0.0);

glTranslatef(0.54, 0.0, 0.0);

glMaterialfv( GL\_FRONT, GL\_AMBIENT, mat7.amb );

glMaterialfv( GL\_FRONT, GL\_DIFFUSE, mat7.dif );

glMaterialfv( GL\_FRONT, GL\_SPECULAR, mat7.spe );

glMaterialf ( GL\_FRONT, GL\_SHININESS, mat7.shi );

glutSolidSphere(0.03, 10, 8);

//Draw the Uranus

glLoadIdentity();

glRotatef((GLfloat)Cheon, 0.0, 1.0, 0.0);

glTranslatef(0.6, 0.0, 0.0);

glMaterialfv( GL\_FRONT, GL\_AMBIENT, mat8.amb );

glMaterialfv( GL\_FRONT, GL\_DIFFUSE, mat8.dif );

glMaterialfv( GL\_FRONT, GL\_SPECULAR, mat8.spe );

glMaterialf ( GL\_FRONT, GL\_SHININESS, mat8.shi );

glutSolidSphere(0.02, 10, 8);

//Draw the Neptune

glLoadIdentity();

glRotatef((GLfloat)Hae, 0.0, 1.0, 0.0);

glTranslatef(0.65, 0.0, 0.0);

glMaterialfv( GL\_FRONT, GL\_AMBIENT, mat9.amb );

glMaterialfv( GL\_FRONT, GL\_DIFFUSE, mat9.dif );

glMaterialfv( GL\_FRONT, GL\_SPECULAR, mat9.spe );

glMaterialf ( GL\_FRONT, GL\_SHININESS, mat9.shi );

glutSolidSphere(0.01, 10, 8);

glutSwapBuffers();//Swap background buffer

}

void timer(int t){

//Represent Each planet around the sun by different speed.

Su = (Su + 18) % 360;

Geom = (Geom + 13) % 360;

Ji\_Day = (Ji\_Day + 11) % 360;

Ji\_Time = (Ji\_Time + 7) % 360;

Hwa = (Hwa + 8) % 360;

Mok = (Mok + 6) % 360;

To = (To + 5) % 360;

Cheon = (Cheon + 4) % 360;

Hae = (Hae + 3) % 360;

glutPostRedisplay(); //Dispaly function call.

glutTimerFunc(delay, timer, t);

}

int main(int argc, char\*\* argv) {

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH);

glutInitWindowSize(500, 500);

glutInitWindowPosition(0, 0);

glutCreateWindow("Solar System");

glMatrixMode(GL\_PROJECTION);

glClearColor(0.0, 0.0, 0.0, 0.0);

glutTimerFunc(delay, timer, 0);

glutDisplayFunc(MyDisplay);

//init light

glLightfv( GL\_LIGHT0, GL\_POSITION, light.pos );

glLightfv( GL\_LIGHT0, GL\_AMBIENT, light.amb );

glLightfv( GL\_LIGHT0, GL\_DIFFUSE, light.dif );

glLightfv( GL\_LIGHT0, GL\_SPECULAR, light.spe );

//Set light effect

glShadeModel( GL\_SMOOTH );

glEnable ( GL\_LIGHTING );

glEnable ( GL\_LIGHT0 );

glutMainLoop();

return 0;

}