#include <GL/glut.h> // Header File For The GLUT Library

float rt = 0.0f;

void init(int Width, int Height)

{

// This Will Clear The Background Color To Black

glClearColor(0.0f, 0.0f, 0.0f, 0.0f);

/\* two matrix modes- GL\_PROJECTION (for setting the projection transformation) &

GL\_MODELVIEW (for setting the modeling and viewing transformations) \*/

glMatrixMode(GL\_PROJECTION);

gluPerspective(45.0f,(GLfloat)Width/(GLfloat)Height,0.1f,50.0f); //View angle,aspect-ratio, viewer near,far distance from clipping plane

glMatrixMode(GL\_MODELVIEW);

}

float ballX = -0.5f;

float ballY = 0.0f;

float ballZ = 0.0f;

/\* The main drawing function. \*/

void Draw()

{

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT); // Clear The Screen And The Depth Buffer

glLoadIdentity(); // Reset The View..Replace the current matrix with the identity matrix

glTranslatef(rt,0.0f,-6.0f); // used to Move scene forward ...

glBegin(GL\_POLYGON); // start drawing a polygon

glColor3f(1.0f,0.0f,0.0f); // Set The Color To Red

glVertex3f(-1.0f, 1.0f, 0.0f); // Top left

glVertex3f(0.4f, 1.0f, 0.0f);

glVertex3f(1.0f, 0.4f, 0.0f);

glColor3f(0.0f,1.0f,0.0f); // Set The Color To Green

glVertex3f( 1.0f,0.0f, 0.0f); // Bottom Right

glColor3f(0.0f,0.0f,1.0f); // Set The Color To Blue

glVertex3f(-1.0f,0.0f, 0.0f);// Bottom Left

glEnd();

glColor3f(0.0, 1.0, 0.0); //set ball colour

glTranslatef(ballX,ballY,ballZ); //moving it toward the screen a bit on creation

glutSolidSphere (0.3, 20, 20); //create ball. radius,longitude,latitute

glTranslatef(ballX+1.5,ballY,ballZ); //moving it toward the screen a bit on creation

glutSolidSphere (0.3, 20, 20);

rt+=0.005f; // Increase The Rotation Variable For The Triangle

if(rt>2) //Set to left corner

rt=-2.0f;

glutSwapBuffers(); // swap the buffers to display, since double buffering is used.

}

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

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_RGBA | GLUT\_SINGLE ); // GLUT\_SINGLE fast movement

glutInitWindowSize(640, 480);

glutInitWindowPosition(0, 0);

glutCreateWindow("Moving Car");

glutDisplayFunc(Draw);

glutIdleFunc(Draw); //global idle callback for continuous animation

/\* Initialize our window. \*/

init(640,480);

/\* Start Event Processing Engine \*/

glutMainLoop();

return 0;

}