|  |
| --- |
| #include <GL/glut.h> |
|  | #include <math.h> |
|  | #include <iostream> |
|  |  |
|  | using namespace std; |
|  | float inc = 1.0; |
|  | float angle = 135; |
|  | float drawCircle(float segments, float radius, float sx, float sy) { |
|  | glBegin(GL\_LINE\_LOOP); |
|  | for (int i = 0;i<segments; i++) { |
|  | float theta = 2.0 \* 3.142 \* float(i) / float(segments); |
|  | // get the current angle |
|  | float x = radius \* cos(theta); |
|  | float y = radius \* sin(theta); |
|  | glVertex2f(x + sx, y + sy); |
|  | } |
|  | glEnd(); |
|  | } |
|  | void draw(float x1, float y1, float angle) { |
|  | float segments = 100; |
|  | float radius = 6.0; |
|  | // Drawing Clock main Circle |
|  | glLineWidth(4); |
|  | glColor3f(1, 0, 0); |
|  | drawCircle(segments, radius, x1, y1); |
|  | // Drawing Minute Line |
|  | glColor3f(1, 1, 0); |
|  | glLineWidth(2); |
|  | glBegin(GL\_LINES); |
|  | glVertex2f(x1, y1); |
|  |  |
|  | glVertex2f(x1, (radius / 3.0) \* 2.0); |
|  | glEnd(); |
|  | // Drawing Hour Line |
|  | glColor3f(1, 0, 0); |
|  | glLineWidth(2); |
|  | glBegin(GL\_LINES); |
|  | glVertex2f(x1, y1); |
|  | glVertex2f(radius / 3.0, radius / 3.0); |
|  | glEnd(); |
|  | // Drawing Pendulum Circle |
|  | double radians = angle \* 3.142 / 180; |
|  | float x2 = (radius \* 3.4) \* cos(radians); |
|  | float y2 = (radius \* 3.4) \* sin(radians); |
|  | float radius2 = radius / 3.0; |
|  | glColor3f(0, 0, 1); |
|  | // glLineWidth(2); |
|  | drawCircle(segments, radius2, x2, y2); |
|  | glBegin(GL\_LINES); |
|  | glVertex2f(x1, -1 \* (radius) + 0.2); |
|  | glVertex2f(x2, y2); |
|  | glEnd(); |
|  | } |
|  | void display() { |
|  | glClearColor(0, 0, 0, 1); |
|  | glClear(GL\_COLOR\_BUFFER\_BIT); |
|  | glLoadIdentity(); |
|  | glTranslatef(-10, 10, -30); |
|  | glColor3f(1, 1, 1); |
|  | if (angle > 315) { |
|  | angle = 315; |
|  | inc = -inc; |
|  | } |
|  | Sleep(40); |
|  | if (angle < 225) { |
|  | angle = 225; |
|  | inc = -inc; |
|  | } |
|  | angle += inc; |
|  | draw(0, 0, angle); |
|  | glutSwapBuffers(); |
|  | } |
|  | void reshape(int w, int h) { |
|  | glMatrixMode(GL\_PROJECTION); // maps camera to screen |
|  | glLoadIdentity(); |
|  | gluPerspective(100, (GLfloat)w / (GLfloat)h, 0.5, 100.0); |
|  | glMatrixMode(GL\_MODELVIEW); |
|  | } |
|  | int main(int argc, char\*\* argv) { |
|  | glutInit(&argc, argv); |
|  | glutInitDisplayMode(GLUT\_DOUBLE); |
|  | glutInitWindowSize(800, 600); |
|  | glutInitWindowPosition(0, 0); |
|  | glutCreateWindow("animation"); |
|  | glutDisplayFunc(display); |
|  | glutIdleFunc(display); |
|  | glutReshapeFunc(reshape); |
|  | glutMainLoop(); |
|  | return 0; |
|  | } |