**10.**

#include <stdlib.h>

#include <GL/glut.h>

#include<gl\GL.h>

#include<gl\GLU.h>

#include <time.h>

GLfloat vertices[] = { -1.0,-1.0,-1.0,1.0,-1.0,-1.0,

1.0,1.0,-1.0, -1.0,1.0,-1.0, -1.0,-1.0,1.0,

1.0,-1.0,1.0, 1.0,1.0,1.0, -1.0,1.0,1.0 };

GLfloat normals[] = { -1.0,-1.0,-1.0,1.0,-1.0,-1.0,

1.0,1.0,-1.0, -1.0,1.0,-1.0, -1.0,-1.0,1.0,

1.0,-1.0,1.0, 1.0,1.0,1.0, -1.0,1.0,1.0 };

GLfloat colors[] = { 0.0,0.0,0.0,1.0,0.0,0.0,

1.0,1.0,0.0, 0.0,1.0,0.0, 0.0,0.0,1.0,

1.0,0.0,1.0, 1.0,1.0,1.0, 0.0,1.0,1.0 };

GLubyte cubeIndices[] = { 0,3,2,1,2,3,7,6,0,4,7,3,1,2,6,5,4,5,6,7,0,1,5,4 };

static GLfloat theta[] = { 0.0,0.0,0.0 };

static GLfloat beta[] = { 0.0,0.0,0.0 };

static GLint axis = 2;

void delay(float secs)

{

float end = clock() / CLOCKS\_PER\_SEC + secs;

while ((clock() / CLOCKS\_PER\_SEC) < end);

}

void displaySingle(void)

{

/\* display callback, clear frame buffer and z buffer,

rotate cube and draw, swap buffers \*/

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glLoadIdentity();

glRotatef(theta[0], 1.0, 0.0, 0.0);

glRotatef(theta[1], 0.0, 1.0, 0.0);

glRotatef(theta[2], 0.0, 0.0, 1.0);

glDrawElements(GL\_QUADS, 24, GL\_UNSIGNED\_BYTE, cubeIndices);

glBegin(GL\_LINES);

glVertex3f(0.0, 0.0, 0.0);

glVertex3f(1.0, 1.0, 1.0);

glEnd();

glFlush();

}

void spinCube()

{

/\* Idle callback, spin cube 2 degrees about selected axis \*/

//sleep(50);

delay(0.01);

theta[axis] += 2.0;

if (theta[axis] > 360.0) theta[axis] -= 360.0;

glutPostRedisplay();

}

void mouse(int btn, int state, int x, int y)

{

/\* mouse callback, selects an axis about which to rotate \*/

if (btn == GLUT\_LEFT\_BUTTON && state == GLUT\_DOWN) axis = 0;

if (btn == GLUT\_MIDDLE\_BUTTON && state == GLUT\_DOWN) axis = 1;

if (btn == GLUT\_RIGHT\_BUTTON && state == GLUT\_DOWN) axis = 2;

}

void myReshape(int w, int h)

{

glViewport(0, 0, w, h);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

if (w <= h)

glOrtho(-2.0, 2.0, -2.0 \* (GLfloat)h / (GLfloat)w,

2.0 \* (GLfloat)h / (GLfloat)w, -10.0, 10.0);

else

glOrtho(-2.0 \* (GLfloat)w / (GLfloat)h,

2.0 \* (GLfloat)w / (GLfloat)h, -2.0, 2.0, -10.0, 10.0);

glMatrixMode(GL\_MODELVIEW);

}

void

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

{

//window 1

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowPosition(100, 100);

glutInitWindowSize(500, 500);

glutCreateWindow("colorcube");

glutReshapeFunc(myReshape);

glutDisplayFunc(displaySingle);

glutIdleFunc(spinCube);

glutMouseFunc(mouse);

glEnable(GL\_DEPTH\_TEST); /\* Enable hidden--surface--removal \*/

glEnableClientState(GL\_COLOR\_ARRAY);

glEnableClientState(GL\_NORMAL\_ARRAY);

glEnableClientState(GL\_VERTEX\_ARRAY);

glVertexPointer(3, GL\_FLOAT, 0, vertices);

glColorPointer(3, GL\_FLOAT, 0, colors);

glNormalPointer(GL\_FLOAT, 0, normals);

glColor3f(1.0, 1.0, 1.0);

glutMainLoop();

}

**OUTPUT:**



