OpenGl example: A simple red drawn with minimal OpenGL calls.

#include <GL/glut.h>

GLfloat light\_diffuse[] = {1.0, 0.0, 0.0, 1.0};  /\* Red diffuse light. \*/

GLfloat light\_position[] = {1.0, 1.0, 1.0, 0.0};  /\* Infinite light location. \*/

GLfloat n[6][3] = {  /\* Normals for the 6 faces of a cube. \*/

  {-1.0, 0.0, 0.0}, {0.0, 1.0, 0.0}, {1.0, 0.0, 0.0},

  {0.0, -1.0, 0.0}, {0.0, 0.0, 1.0}, {0.0, 0.0, -1.0} };

GLint faces[6][4] = {  /\* Vertex indices for the 6 faces of a cube. \*/

  {0, 1, 2, 3}, {3, 2, 6, 7}, {7, 6, 5, 4},

  {4, 5, 1, 0}, {5, 6, 2, 1}, {7, 4, 0, 3} };

GLfloat v[8][3];  /\* Will be filled in with X,Y,Z vertexes. \*/

void

drawBox(void)

{

  int i;

  for (i = 0; i < 6; i++) {

    glBegin(GL\_QUADS);

    glNormal3fv(&n[i][0]);

    glVertex3fv(&v[faces[i][0]][0]);

    glVertex3fv(&v[faces[i][1]][1]);

    glVertex3fv(&v[faces[i][2]][0]);

    glVertex3fv(&v[faces[i][3]][0]);

    glEnd();

  }

}

void

display(void)

{

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

  drawBox();

  glutSwapBuffers();

}

void

init(void)

{

  /\* Setup cube vertex data. \*/

  v[0][0] = v[1][0] = v[2][0] = v[3][0] = -1;

  v[4][0] = v[5][0] = v[6][0] = v[7][0] = 1;

  v[0][1] = v[1][1] = v[4][1] = v[5][1] = -1;

  v[2][1] = v[3][1] = v[6][1] = v[7][1] = 1;

  v[0][2] = v[3][2] = v[4][2] = v[7][2] = 1;

  v[1][2] = v[2][2] = v[5][2] = v[6][2] = -1;

  /\* Enable a single OpenGL light. \*/

  glLightfv(GL\_LIGHT0, GL\_DIFFUSE, light\_diffuse);

  glLightfv(GL\_LIGHT0, GL\_POSITION, light\_position);

  glEnable(GL\_LIGHT0);

  glEnable(GL\_LIGHTING);

  /\* Use depth buffering for hidden surface elimination. \*/

  glEnable(GL\_DEPTH\_TEST);

  /\* Setup the view of the cube. \*/

  glMatrixMode(GL\_PROJECTION);

  gluPerspective( /\* field of view in degree \*/ 40.0,

    /\* aspect ratio \*/ 1.0,

    /\* Z near \*/ 1.0, /\* Z far \*/ 10.0);

  glMatrixMode(GL\_MODELVIEW);

  gluLookAt(0.0, 0.0, 5.0,  /\* eye is at (0,0,5) \*/

    0.0, 0.0, 0.0,      /\* center is at (0,0,0) \*/

    0.0, 1.0, 0.);      /\* up is in positive Y direction \*/

  /\* Adjust cube position to be asthetic angle. \*/

  glTranslatef(0.0, 0.0, -1.0);

  glRotatef(60, 1.0, 0.0, 0.0);

  glRotatef(-20, 0.0, 0.0, 1.0);

}

int

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

{

  glutInit(&argc, argv);

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

  glutCreateWindow("red 3D lighted cube");

  glutDisplayFunc(display);

  init();

  glutMainLoop();

  return 0;             /\* ANSI C requires main to return int. \*/

}

OutPut:

