Program 10

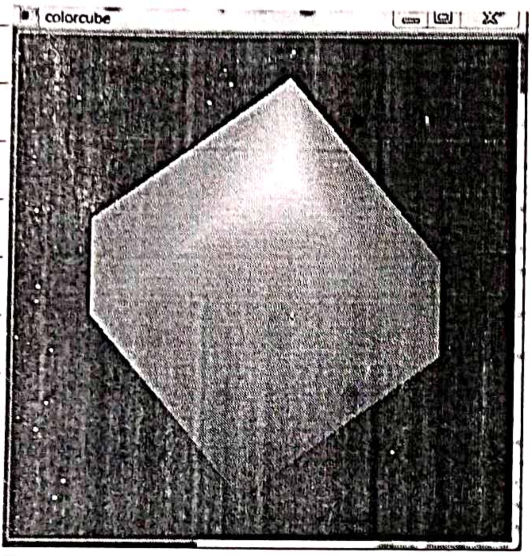
Write a program to create a color cube and spin it using OpenGL transformations.

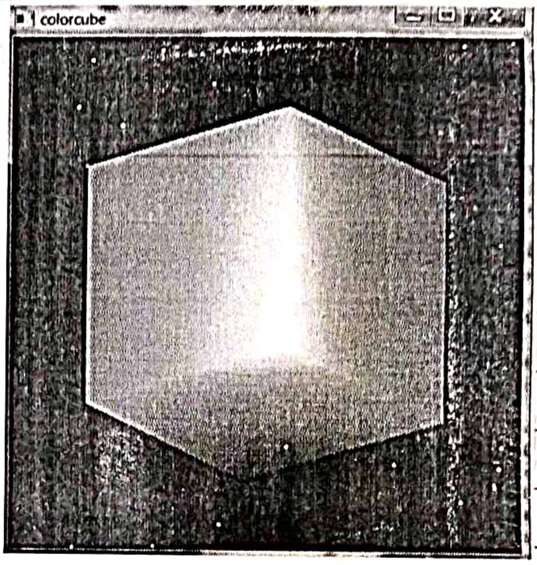
```
#include <stdlib.h>
#include <GL/glut.h>
#include <gl/GL.h>
#include <gl/GLUT.h>
# include<time.h>

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};

GLfoat 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};

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

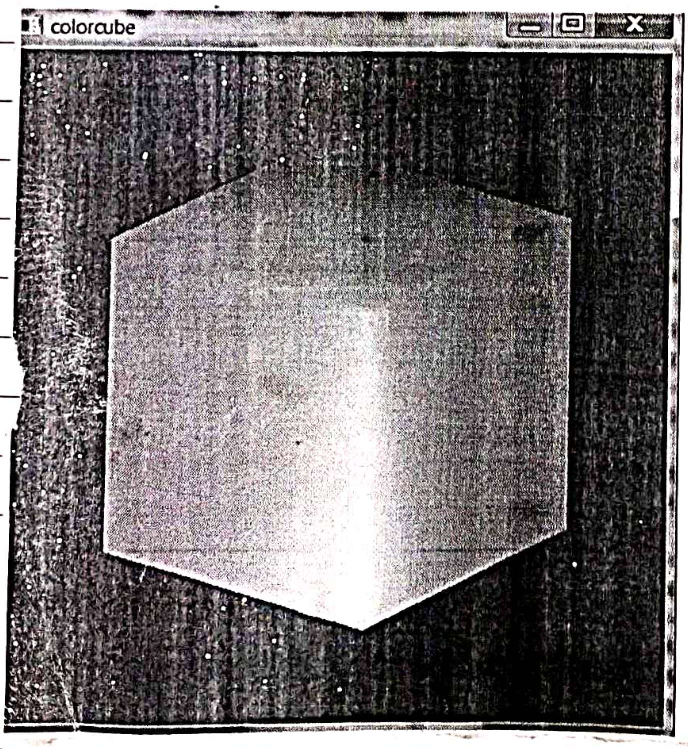
```
void delay (float secs)
{
    float end = clock() / CLOCKS_PER_SEC + secs;
    while ((clock() / CLOCDS_PER_SEC) < end);
}

void desplaySingle (void)
{
    glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glLoad Identity ();
    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()
{
      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)
{
    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 myResphape(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 agrc, char** argv)
{
        glutInit(&argc, argv);
        glutInitDisplayMode (GLUT_SINGLE |GLUT_RGB);
        glutMouseFunc(mouse);
      glEnable (GL_DEPTH_TEST);
      glEnableClientState(GL_COLOR_ARRAY);
      glEnableClientState(GL_NORMAL_ARRAY);
     glEnableClientState( GL_VERTEX_ARRAY);
     glVertex pointer (3, GL_FLOAT, 0, vertices);
     glcolor pointer (3, GL_FLOAT, 0, colors);
     glNormal pointer ( GL_FLOAT, 0, normals);
     glcolor3f (1.0,1.0,1.0);
     glutMainLoop();

}
```


colorcube


colorcube


colorcube