```
/**
   Example program 4.
   Animate two rotateing cubes, using the idle callback and double
   buffering.
   Also shows how to read the mouse buttons.

   @author Lukas Ahrenberg (lukas@madscilab.com).
 */

#include <GL/gl.h>
#include <GL/glut.h>
#include <stdlib.h>
#include <stdio.h>

/*--- CONSTANTS ---*/
// Title.
const char* TITLE = "Program 4";

/*--- GLOBALS ---*/

// The vertices.
GLfloat vertices[][3] = {{-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}};

// The Normals of each vertex.
GLfloat normals[][3] = {{-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}};

// The colours of each vertex.
GLfloat colors[][3] = {{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}};

// The current rotation axis index.
GLint axis = 0;

// Array containing the rotation angle around each of the axis x,y,z.
GLfloat theta[] = {0.0, 0.0, 0.0};


/**
   GLUT init function. Initializes GLUT, sets up display mode and opens
   a window. Needs the args for the glutInit.
   @param argc Argument count.
   @param argv Argument array.
 */
void initGLUT( int &argc, char **argv)
{
  glutInit( &argc,argv );
  glutInitDisplayMode( GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH );
  glutInitWindowSize( 700, 700 );
  glutInitWindowPosition( 100, 100 );
  glutCreateWindow( TITLE );
}

/**
   GL init function. Sets background colour and projection.
 */
void initGL()
{
  glClearColor( 0.0, 0.0, 0.0, 0.0 );

  // Setting up camera insrinct parameters.
  glMatrixMode( GL_PROJECTION );
```

```
  glLoadIdentity();
  gluPerspective(65.0, 1.0, 1.0, 20.0 );

  // Setting up camera extrinct parameters.
  glMatrixMode (GL_MODELVIEW );
  glLoadIdentity();
  gluLookAt(1,-4,-4,0,0,0,0,1,0);

  glEnable(GL_DEPTH_TEST);
  glShadeModel( GL_SMOOTH );
}


/**
    Defines a quadratic polygon between vertices a,b,c,d.
    @param a,b,c,d The vertices.
*/
void polygon(int a, int b, int c , int d)
{
        glBegin(GL_POLYGON);
        glColor3fv(colors[a]);
        glNormal3fv(normals[a]);
        glVertex3fv(vertices[a]);

        glColor3fv(colors[b]);
        glNormal3fv(normals[b]);
        glVertex3fv(vertices[b]);

        glColor3fv(colors[c]);
        glNormal3fv(normals[c]);
        glVertex3fv(vertices[c]);

        glColor3fv(colors[d]);
        glNormal3fv(normals[d]);
        glVertex3fv(vertices[d]);
        glEnd();
}

/**
    Build a cube using the polygon function.
*/
void colorCube(void)
{
  // Draw the different sides.
        polygon(0,3,2,1);
        polygon(2,3,7,6);
        polygon(0,4,7,3);
        polygon(1,2,6,5);
        polygon(4,5,6,7);
        polygon(0,1,5,4);
}



/**
   Draw function. Defines three vertices and draws a polygon.
 */
void draw()
{
  glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
  glColor3f(1.0, 1.0, 1.0);

  // Big cube.
  glPushMatrix();
  glRotatef(theta[0],1,0,0);
  glRotatef(theta[1],0,1,0);
```

```
  glRotatef(theta[2],0,0,1);
  colorCube();
  glPopMatrix();

  // Small block.
  glPushMatrix();
  glTranslatef(2,1,0);
  glRotatef(45,1,0,1);
  glScalef(0.2,0.5,0.5);
  colorCube();
  glPopMatrix();

  glFlush();

  glutSwapBuffers();
}

/**
   Animation callback.
   Increase angle of rotation for current axis.
 */
void animate()
{
  theta[axis] +=2.0;
  if(theta[axis]>360.0)
    theta[axis]-=360.0;
  glutPostRedisplay();
}

/**
   Mouse callback.
   Check which button has been pressed and set appropriate axis.
   @param btn The putton pressed, if any.
   @param state Event information about the mouse.
   @param x,y Position of the cursor.
 */
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;
  }
}

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

  initGLUT(argc, argv);
  initGL();

  // Set the callback functions.
  glutDisplayFunc( draw );
  glutIdleFunc( animate );
  glutMouseFunc(mouse);
  //Start the GLUT main Loop.
  glutMainLoop();

  return 0;
}
```