

Cilindro

```
#include <GL/glut.h>

GLUquadricObj *qobj;

//GLfloat girax = 15, giray= 0, zoom= 0; //Ortogonal
GLfloat girax = 0, giray= 0, zoom= -20; //Perspectiva
GLboolean showGrid = GL_FALSE;
GLboolean showAxes = GL_FALSE;

/* ----- Rotacion y Zoom ----- */
void mover (void)
{
    glTranslated( 0, 0, zoom);
    glRotated( girax, 1.0, 0.0, 0.0);
    glRotated( giray, 0.0, 1.0, 0.0);
}

/* ----- Rejilla y Ejes ----- */
GLvoid Grid (GLvoid)
{
    GLfloat x,z;
    float long_eje= 10;

    glColor3f(1.0,0.0,0.0);
    glLineWidth(2);
    glBegin(GL_LINES);
    for (x = -long_eje; x<=long_eje; x+= 0.5 ) {
        glVertex3f (x,0,-long_eje);
        glVertex3f (x,0,long_eje);
    }
    for (z = -long_eje; z<= long_eje; z+= 0.5 ) {
        glVertex3f (-long_eje,0,z);
        glVertex3f (long_eje,0,z);
    }
    glEnd();
}

GLvoid Ejes ( GLvoid)
{
    glColor3f(1.0,1.0,0.0); /* amarillo */
    glBegin( GL_LINES );
    glVertex3f (0.0,0.0,0.0);
    glVertex3f (11.0,0.0,0.0);
    glVertex3f (0.0,0.0,0.0);
    glVertex3f (0.0,11.0,0.0);
    glVertex3f (0.0,0.0,0.0);
    glVertex3f (0.0,0.0,11.0);
    glEnd();
    glBegin( GL_TRIANGLES );
    glVertex3f (11.0,0.0,0.0); /* eje x */
    glVertex3f (10.5,0.0,-.50);
    glVertex3f (10.5,0.0,.50);
    glVertex3f (0.0,11.0,0.0); /* eje y */
    glVertex3f (-.50,10.5,0.0);
    glVertex3f (.50,10.5,0.0);
    glColor3f(1.0,0.0,1.0); /* lila eje z*/
    glVertex3f (0.0,0.0,11.0);
    glVertex3f (-.50,0.0,10.5);
    glVertex3f (.50,0.0,10.5);
    glEnd();
}

/* ----- Dibuja Escena ----- */
void dibuja(void)
{
    qobj = gluNewQuadric ();

    glClear(GL_COLOR_BUFFER_BIT);

    glPushMatrix();
    mover();
    if ( showGrid == GL_TRUE ) Grid();
    if ( showAxes == GL_TRUE ) Ejes();
    glColor3f(.2,.2,1.);
    gluCylinder(qobj,1,1,8,20,5);
    glPopMatrix();

    glutSwapBuffers();
}
```

```

/* ----- Funciones con Teclas ----- */
void keyboard(unsigned char key, int x, int y) {
    switch(key) {
        case 27: exit (0);
                break;
        case 'i': zoom += 1;
                break;
        case 'o': zoom -= 1;
                break;
        case 'g': showGrid = !showGrid;
                break;
        case 'e': showAxes = !showAxes;
                break;
        default: break;
    }
    glutPostRedisplay();
}

void flechas(int key, int x, int y) {
    switch(key) {
        case GLUT_KEY_LEFT: giray-= 15;
                break;
        case GLUT_KEY_RIGHT: giray+= 15;
                break;
        case GLUT_KEY_UP: girax -= 15;
                break;
        case GLUT_KEY_DOWN: girax += 15;
                break;
        default:break;
    }
    glutPostRedisplay();
}

/* ----- Reshape ----- */
void reshape(int width, int height)
{
    glClearColor(1.0,1.0,1.0,0.0);
    glViewport(0, 0, width, height);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(45, (float)width/height,0,80);
    //glOrtho(-20,20,-20,20,-20,20);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
}

/* ----- main ----- */
int main(int argc, char** argv)
{
    glutInitDisplayMode(GLUT_RGB|GLUT_DOUBLE);
    glutInitWindowPosition(200, 200);
    glutInitWindowSize(640, 480);
    glutInit(&argc, argv);
    glutCreateWindow("Cilindro");

    glutDisplayFunc(dibuja);
    glutReshapeFunc(reshape);
    glutKeyboardFunc(keyboard);
    glutSpecialFunc(flechas);

    glutMainLoop();
    return 0;
}

```

