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


GLsizei W = 1000, H = 1000;
GLfloat Rx = 0.0, Ry = 0.0;
void Key(int key, int x, int y);
void Cube(GLfloat x, GLfloat y, GLfloat z, GLfloat l);
void Cube(GLfloat x, GLfloat y, GLfloat z, GLfloat l) {
	GLfloat h = l * 0.5;
	glRotatef(Rx, 1, 0, 0);
	glRotatef(Ry, 0, 1, 0);
	GLfloat V[] =
	{
		//FRONT

		x - h,y + h,z + h,
		x + h,y + h,z + h,
		x + h,y - h,z + h,
		x - h,y - h,z + h,

		//BACK
		x - h,y + h,z - h,
		x + h,y + h,z - h,
		x + h,y - h,z - h,
		x - h,y - h,z - h,
		//LEFT
		x - h,y + h,z + h,
		x - h,y + h,z - h,
		x - h,y - h,z - h,
		x - h,y - h,z + h,
		//RIGHT
		x + h,y + h,z + h,
		x + h,y + h,z - h,
		x + h,y - h,z - h,
		x + h,y - h,z + h,
		//TOP
		x - h,y + h,z + h,
		x - h,y + h,z - h,
		x + h,y + h,z - h,
		x + h,y + h,z + h,
		//BOTTOM
		x - h,y - h,z + h,
		x - h,y - h,z - h,
		x + h,y - h,z - h,
		x + h,y - h,z + h,
	};
	glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);
	glEnableClientState(GL_VERTEX_ARRAY);
	glVertexPointer(3, GL_FLOAT, 0, V);
	glDrawArrays(GL_QUADS, 0, 24);
	glDisableClientState(GL_VERTEX_ARRAY);
}
void Key(int key, int x, int y) {
	if (key == GLUT_KEY_RIGHT) {
		Ry += 4;
	}
```

```
	else if (key == GLUT_KEY_LEFT) {
		Ry -= 4;
	}
	else if (key == GLUT_KEY_UP) {
		Rx += 4;
	}
	else if (key == GLUT_KEY_DOWN) {
		Rx -= 4;
	}
	glutPostRedisplay();
}
void start() {
	glViewport(0.0, 0.0, W, H);
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	glOrtho(0 ,W, 0, H, 0, 100000000000000000);
	glMatrixMode(GL_MODELVIEW);
	glLoadIdentity();
}
void display() {
	glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
	glPushMatrix();
	Cube((W / 2), (H / 2), -500, 200);
	glPopMatrix();
	glutSwapBuffers();
}
int main(int argc, char* argv[]) {
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
	glutInitWindowSize(W, H);
	glutCreateWindow("Cubo");
	start();
	glutDisplayFunc(display);
	glutSpecialFunc(Key);
	glutMainLoop();
	return 0;
}
```