3. Program to draw a color cube and spin it using openGL transformation matrices.

Objective: In this program the students will learn to draw a color cube and spin it using OpenGL functions.

Program

```
//lab3:Spin color cube
#include<windows.h>
#include <GL/glut.h>

GLfloat vertices[8][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}};

GLfloat colors[8][3] = { {0.0,0.0,1.0}, {0.0,1.0,1.0}, {1.0,1.0,1.0}, {1.0,0.0,1.0},
{0.0,0.0,0.0},{0.0,1.0,0.0}, {1.0,1.0,0.0}, {1.0,0.0,0.0}};

void polygon(int a, int b, int c , int d)
{
/* draw a polygon via list of vertices */
glBegin(GL_POLYGON);
glColor3fv(colors[a]);
glVertex3fv(vertices[a]);
glColor3fv(colors[b]);
glVertex3fv(vertices[b]);
glColor3fv(colors[c]);
glVertex3fv(vertices[c]);
glColor3fv(colors[d]);
glVertex3fv(vertices[d]);
glEnd();
}
/* map vertices to faces */
void colorcube()
{
polygon(0,3,2,1); // front face – counter clockwise
polygon(4,5,6,7); // back face – clockwise

polygon(2,3,7,6); // front face – counter clockwise
polygon(1,5,4,0); // back face – clockwise

polygon(1,2,6,5); // front face – counter clockwise
polygon(0,4,7,3); // back face – clockwise
}
static GLfloat theta[] = {0.0,0.0,0.0};
static GLint axis = 2; /* default z-axis rotation – 0(x axis), 1(y axis), 2(z axis)*/

void display(void)
{
/* display callback, clear frame buffer and z buffer, rotate cube and draw, swap buffers */
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
glLoadIdentity();
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);
colorcube();
glFlush();
glutSwapBuffers();
}
/* Idlecallback, spin cube about selected axis */
void spinCube()
{
theta[axis] += 0.05; // Controls the speed
if( theta[axis] > 360.0 ) theta[axis] -= 360.0;
glutPostRedisplay();
}
/* mouse callback, selects an axis about which to rotate */
void mouse(int btn, int state, int x, int y)
{
/* mouse callback, selects an axis about which to rotate */
        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 myReshape(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);
}

int main(int argc, char **argv)
{
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
        /* need both double buffering and z buffer */
        glutInitWindowSize(500, 500);
        glutCreateWindow("Rotating a Color Cube");
        glutReshapeFunc(myReshape);
        glutDisplayFunc(display);
        glutIdleFunc(spinCube);
        glutMouseFunc(mouse);
        glEnable(GL_DEPTH_TEST); /* Enable hidden--surface--removal */
        glutMainLoop();
         return 0;
}
```

Output: