

Name : Vaishnavi Bharat Gangurde

Roll No : 25

Division :A

Assignment 2_6_2 : OpenGL

SOURCE CODE:

```
//Assignment 10
/* Problem Statement: Write C++ program to draw 3-D cube
and perform rotation on it using OpenGL. */

#include <GL/glut.h> // GLUT, include glu.h and gl.h

/* Global variables */
GLfloat angleCube = 0.0f; // Rotational angle for
cube [NEW] int refreshMills = 15;
// refresh interval in milliseconds [NEW]

/* Initialize OpenGL Graphics */

void initGL() {
    glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
    /* Set background color to black and opaque*/

    glClearDepth(1.0f);
    /* Set background depth to farthest*/

    glEnable(GL_DEPTH_TEST);
    /* Enable depth testing for z-culling*/
}

/* Handler for window-repaint event. Called back when the
window first appears and whenever the window needs to be
re-painted. */

void display()
{
    glClear(GL_COLOR_BUFFER_BIT |
GL_DEPTH_BUFFER_BIT); /*Clear color and
depth buffers */

    glMatrixMode(GL_MODELVIEW);
    /* To operate on model-view matrix*/

    /* Render a color-cube consisting of 6 quads with different
colors*/
```

```

glLoadIdentity();
/* Reset the model-view matrix */

glTranslatef(1.5f, 0.0f, -7.0f);
/* Move right and into the screen */

glRotatef(angleCube, 1.0f, 1.0f, 1.0f);
/* Rotate about (1,1,1)-axis [NEW] */

glBegin(GL_QUADS);
/* Begin drawing the color cube with 6 quads */

/* Top face (y = 1.0f) */
/* Define vertices in counter-clockwise (CCW) order with
normal pointing out */
    glColor3f(0.0f, 1.0f, 0.0f); // Green
    glVertex3f( 1.0f, 1.0f, -1.0f);
    glVertex3f(-1.0f, 1.0f, -1.0f);
    glVertex3f(-1.0f, 1.0f, 1.0f);
    glVertex3f( 1.0f, 1.0f, 1.0f);

// Bottom face (y = -1.0f)
    glColor3f(1.0f, 0.5f, 0.0f); // Orange
    glVertex3f( 1.0f, -1.0f, 1.0f);
    glVertex3f(-1.0f, -1.0f, 1.0f);
    glVertex3f(-1.0f, -1.0f, -1.0f);
    glVertex3f( 1.0f, -1.0f, -1.0f);

// Front face (z = 1.0f)
    glColor3f(1.0f, 0.0f, 0.0f); // Red
    glVertex3f( 1.0f, 1.0f, 1.0f);
    glVertex3f(-1.0f, 1.0f, 1.0f);
    glVertex3f(-1.0f, -1.0f, 1.0f);
    glVertex3f( 1.0f, -1.0f, 1.0f);

// Back face (z = -1.0f)
    glColor3f(1.0f, 1.0f, 0.0f); // Yellow
    glVertex3f( 1.0f, -1.0f, -1.0f);
    glVertex3f(-1.0f, -1.0f, -1.0f);
    glVertex3f(-1.0f, 1.0f, -1.0f);
    glVertex3f( 1.0f, 1.0f, -1.0f);

// Left face (x = -1.0f)
    glColor3f(0.0f, 0.0f, 1.0f); // Blue
    glVertex3f(-1.0f, 1.0f, 1.0f);
    glVertex3f(-1.0f, 1.0f, -1.0f);
    glVertex3f(-1.0f, -1.0f, -1.0f);
    glVertex3f(-1.0f, -1.0f, 1.0f);

// Right face (x = 1.0f)
    glColor3f(1.0f, 0.0f, 1.0f); //
Magenta glVertex3f(1.0f, 1.0f, -1.0f);

```

```

glVertex3f(1.0f, 1.0f, 1.0f);
glVertex3f(1.0f, -1.0f, 1.0f);
glVertex3f(1.0f, -1.0f, -1.0f);
glEnd(); // End of drawing color-cube


glutSwapBuffers();
/* Swap the front and back frame buffers (double
buffering) */ /* Update the rotational angle after
each refresh [NEW] */

    angleCube -= 0.15f;
}

/* Called back when timer expired [NEW] */
void timer(int value) {
    glutPostRedisplay();
    /* Post re-paint request to activate display() */

    glutTimerFunc(refreshMills, timer, 0);

    /* next timer call milliseconds later */
}

/* Handler for window re-size event. Called back when the
window first appears and whenever the window is re-sized with
its new width and height */

void reshape(GLsizei width, GLsizei height)
{
    /* GLsizei for non-negative integer */
    /* Compute aspect ratio of the new window */
    if (height == 0) height = 1;
    /* To prevent divide by 0 */
    GLfloat aspect = (GLfloat)width / (GLfloat)height;

    /* Set the viewport to cover the new window */
    glViewport(0, 0, width, height);

    /* Set the aspect ratio of the clipping volume to match the
viewport */
    glMatrixMode(GL_PROJECTION); /* To operate on the
Projection matrix */
    glLoadIdentity(); // Reset
    /* Enable perspective projection with fovy, aspect, zNear
and zFar */ gluPerspective(45.0f, aspect, 0.1f, 100.0f);
}

```

```

/* Main function: GLUT runs as a console application
starting at main() */
int main(int argc, char** argv) {

    glutInit(&argc, argv); /* Initialize GLUT */

    glutInitDisplayMode(GLUT_DOUBLE); /* Enable double buffered
mode */

    glutInitWindowSize(640, 480);
    /* Set the window's initial width & height */

    glutInitWindowPosition(50, 50);
    /* Position the window's initial top-left corner */

    glutCreateWindow("3D Cube Rotation");
    /* Create window with the given title */

    glutDisplayFunc(display);
    /* Register callback handler for window re-paint

event*/    glutReshapeFunc(reshape);

    /* Register callback handler for window re-size event */

    initGL();
    /* Our own OpenGL initialization */

    glutTimerFunc(0, timer, 0);
    /* First timer call immediately [NEW] */

    glutMainLoop();
    /* Enter the infinite event-processing loop */

    return 0;
}

```

/* Running steps

In Eclipse Environment:

- * Create C/C++ Project
- * Create file Demo.cpp
- * Right Click on Project
- * Click on Properties
- * Select C/C++ Build Tab
- * Click on Settings
- * Select Cross G++ Linker
- * Select Libraries tab.
- * Insert 3 libraries:
 - GL
 - GLU
 - glut

*/