SCHOOL OF COMPUTER SCIENCE

UNIVERSITY OF PETROLEUM AND ENERGY STUDIES
DEHRADUN, UTTARAKHAND

COMPUTER GRAPHICS
LABORATORY FILE
(2024-2025)

For

Vth Semester

Submitted To:
Mr. Dinesh
Assistant Professor
[Vth Semester]
School of Computer Science

Submitted By:
Akshat Negi
500106533(SAP ID)
R2142220414(Roll No.)
B.Tech. CSF (Batch-1)

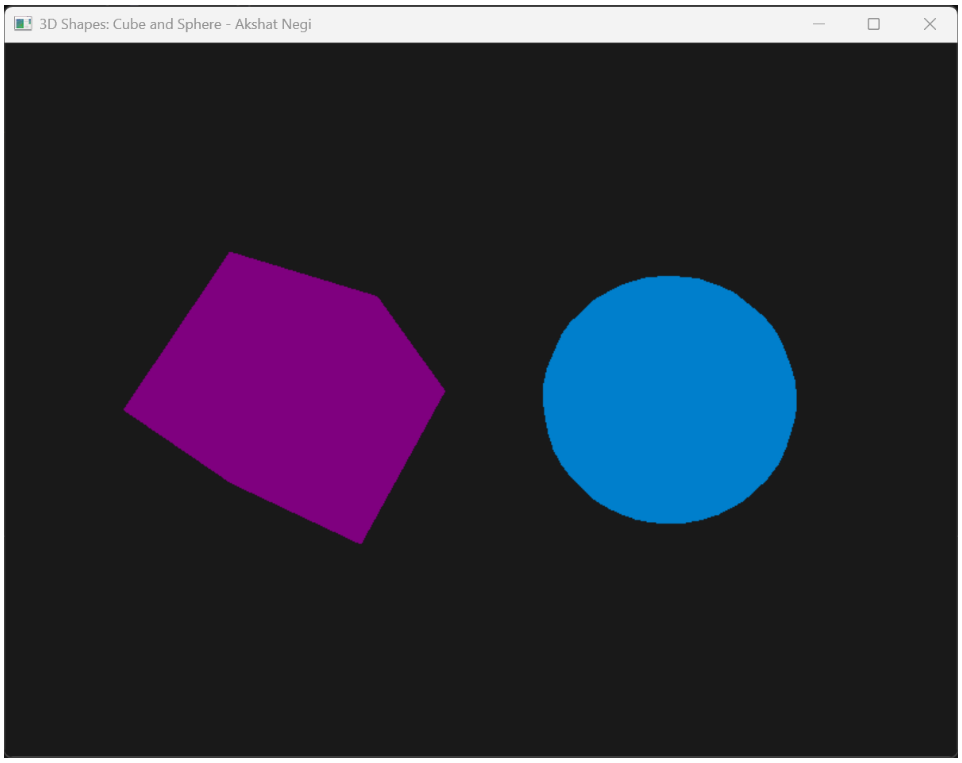
LAB EXPERIMENT – 8

Event Handling

#Implement above with the help of animation.

a. Implement mouse input functionality.
b. Implement keypress functionality.
c. Implement another call back functions.

```
#include <GL/freeglut.h>
#include <iostream>
using namespace std;

float angleCube = 0.0f;  // Rotation angle for the cube
float angleSphere = 0.0f;  // Rotation angle for the sphere
bool rotateCube = true;    // Toggle rotation for the cube
bool rotateSphere = true;  // Toggle rotation for the sphere

// Initialization of OpenGL settings
void initGL() {
    glEnable(GL_DEPTH_TEST);  // Enable depth testing for z-culling
    glClearColor(0.1f, 0.1f, 0.1f, 1.0f); // Set background color to dark gray
}

// Display function to render the shapes
void display() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);  // Clear the screen and
depth buffer
    glMatrixMode(GL_MODELVIEW);  // Switch to the drawing perspective

    // Draw Cube
    glLoadIdentity();
    glTranslatef(-1.5f, 0.0f, -7.0f); // Move left and into the screen
    glRotatef(angleCube, 1.0f, 1.0f, 1.0f); // Rotate the cube
    glColor3f(0.5f, 0.0f, 0.5f);  // Set color of the cube to purple
    glutSolidCube(1.5);  // Draw a cube with side length 1.5

    // Draw Sphere
    glLoadIdentity();
    glTranslatef(1.5f, 0.0f, -7.0f); // Move right and into the screen
    glRotatef(angleSphere, 1.0f, 0.0f, 0.0f); // Rotate the sphere
    glColor3f(0.0f, 0.5f, 0.8f);  // Set color of the sphere to cyan
    glutSolidSphere(1.0, 20, 20);  // Draw a sphere with radius 1.0 and detail
level 20

    glutSwapBuffers();  // Swap front and back buffers (double buffering)
}

// Timer function to update the rotation angles
void timer(int value) {
    if (rotateCube) {
        angleCube += 2.0f;
        if (angleCube > 360) angleCube -= 360;
    }
    if (rotateSphere) {
        angleSphere += 1.5f;
        if (angleSphere > 360) angleSphere -= 360;
    }
    glutPostRedisplay();        // Post a paint request to activate display()
    glutTimerFunc(16, timer, 0); // Call this function again after 16
milliseconds
}

// Keyboard input for rotation toggle
void handleKeypress(unsigned char key, int x, int y) {
    switch (key) {
    case 'c':   // Toggle rotation for the cube
        rotateCube = !rotateCube;
        break;
    case 's':   // Toggle rotation for the sphere
        rotateSphere = !rotateSphere;
        break;
    case 27:    // ESC key
        exit(0);
    }
}

// Reshape function to handle window resizing
void reshape(int width, int height) {
    if (height == 0) height = 1;  // Prevent divide by zero
    float aspect = (float)width / (float)height;

    glViewport(0, 0, width, height);

    // Set the perspective projection
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(45.0f, aspect, 0.1f, 100.0f);
    glMatrixMode(GL_MODELVIEW);
}

int main(int argc, char** argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);  // Enable double
buffering and depth test
    glutInitWindowSize(800, 600);  // Set window size
    glutCreateWindow("3D Shapes: Cube and Sphere - Akshat Negi");  // Create
window with title

    initGL(); // Initialize OpenGL settings
    glutDisplayFunc(display); // Set display function
    glutReshapeFunc(reshape); // Set reshape function
    glutKeyboardFunc(handleKeypress); // Set keyboard input function
    glutTimerFunc(0, timer, 0); // Set timer function

    glutMainLoop(); // Enter the main event loop
    return 0;
}
```


3D Shapes: Cube and Sphere - Akshat Negi