**SCHOOL OF COMPUTER SCIENCE**

**UNIVERSITY OF PETROLEUM AND ENERGY STUDIES**

**DEHRADUN, UTTARAKHAND**



**COMPUTER GRAPHICS**

**LABORATORY FILE**

**(2024-2025)**

**For**

**Vth Semester**

**Submitted To: Submitted By:**

Mr. Dinesh Akshat Negi

Assistant Professor 500106533(SAP ID)

[Vth Semester] R2142220414(Roll No.)

School of Computer Science B.Tech. CSF (Batch-1)

**LAB EXPERIMENT – 8**

**Event Handling**

***#Implement above with the help of animation.***

1. Implement mouse input functionality.
2. Implement keypress functionality.
3. 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

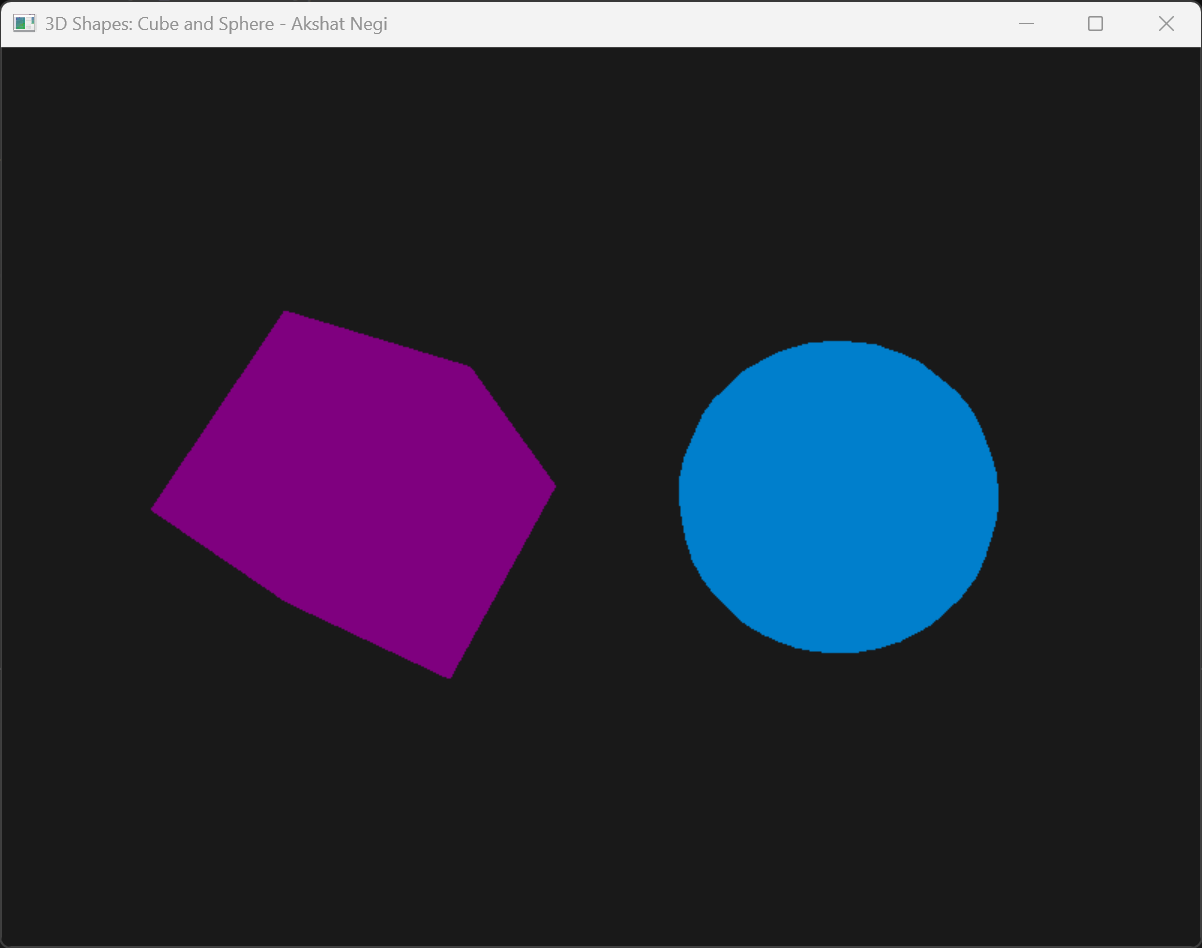
glutKeyboardFunc(handleKeypress); // Set keyboard input function

glutTimerFunc(0, timer, 0); // Set timer function

glutMainLoop(); // Enter the main event loop

return 0;

}

****