Trần Hà Duy – 22IT054 – Lab 07

**Code**

#include <GL/glut.h>

#include <cmath>

#include <algorithm>

float lightPos[3] = {2.0f, 2.0f, 5.0f};

float vertices[4][3] = {

{-1.0f, -1.0f, -1.0f},

{ 1.0f, -1.0f, -1.0f},

{ 1.0f, 1.0f, -1.0f},

{-1.0f, 1.0f, -1.0f}

};

float normals[4][3] = {

{0.0f, 0.0f, 1.0f},

{0.0f, 0.0f, 1.0f},

{0.0f, 0.0f, 1.0f},

{0.0f, 0.0f, 1.0f}

};

float lambertShading(float normal[3], float lightDir[3]) {

float dotProduct = normal[0] \* lightDir[0] + normal[1] \* lightDir[1] + normal[2] \* lightDir[2];

return std::max(0.0f, dotProduct);

}

float gouraudShading(float normal[3], float lightDir[3]) {

float dotProduct = normal[0] \* lightDir[0] + normal[1] \* lightDir[1] + normal[2] \* lightDir[2];

return std::max(0.0f, dotProduct);

}

float phongShading(float normal[3], float lightDir[3], float viewDir[3]) {

float dotLN = normal[0] \* lightDir[0] + normal[1] \* lightDir[1] + normal[2] \* lightDir[2];

float reflect[3] = {

2 \* dotLN \* normal[0] - lightDir[0],

2 \* dotLN \* normal[1] - lightDir[1],

2 \* dotLN \* normal[2] - lightDir[2]

};

float dotRV = reflect[0] \* viewDir[0] + reflect[1] \* viewDir[1] + reflect[2] \* viewDir[2];

float spec = std::max(0.0f, dotRV);

return std::max(0.0f, dotLN) + 0.3f \* std::pow(spec, 10);

}

void drawSphereLambert() {

glColor3f(0.0f, 0.0f, 1.0f);

float normal[3] = {0.0f, 0.0f, 1.0f};

float lightDir[3] = {lightPos[0], lightPos[1], lightPos[2]};

float intensity = lambertShading(normal, lightDir);

glColor3f(intensity, intensity, intensity);

glutSolidSphere(1.0, 36, 18);

}

void drawQuadGouraud() {

glBegin(GL\_QUADS);

for (int i = 0; i < 4; ++i) {

float lightDir[3] = {lightPos[0] - vertices[i][0], lightPos[1] - vertices[i][1], lightPos[2] - vertices[i][2]};

float intensity = gouraudShading(normals[i], lightDir);

glColor3f(intensity, intensity, intensity);

glVertex3fv(vertices[i]);

}

glEnd();

}

void drawQuadPhong() {

glBegin(GL\_QUADS);

for (int i = 0; i < 4; ++i) {

float lightDir[3] = {lightPos[0] - vertices[i][0], lightPos[1] - vertices[i][1], lightPos[2] - vertices[i][2]};

float viewDir[3] = {0.0f - vertices[i][0], 0.0f - vertices[i][1], 5.0f - vertices[i][2]};

float intensity = phongShading(normals[i], lightDir, viewDir);

glColor3f(intensity, intensity, intensity);

glVertex3fv(vertices[i]);

}

glEnd();

}

void initLighting() {

glEnable(GL\_LIGHTING);

glEnable(GL\_LIGHT0);

GLfloat lightPos[] = { 2.0f, 2.0f, 5.0f, 1.0f };

glLightfv(GL\_LIGHT0, GL\_POSITION, lightPos);

GLfloat lightColor[] = { 1.0f, 1.0f, 1.0f, 1.0f };

glLightfv(GL\_LIGHT0, GL\_DIFFUSE, lightColor);

}

void display() {

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glLoadIdentity();

gluLookAt(0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluPerspective(45.0, 1.0, 0.1, 100.0);

glMatrixMode(GL\_MODELVIEW);

//drawSphereLambert(); // Tô bóng Lambert

//drawQuadGouraud(); // Tô bóng Gouraud

drawQuadPhong(); // Tô bóng Phong

glFlush();

glutSwapBuffers();

}

int main(int argc, char\*\* argv) {

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH);

glutCreateWindow("Shading Techniques");

glEnable(GL\_DEPTH\_TEST);

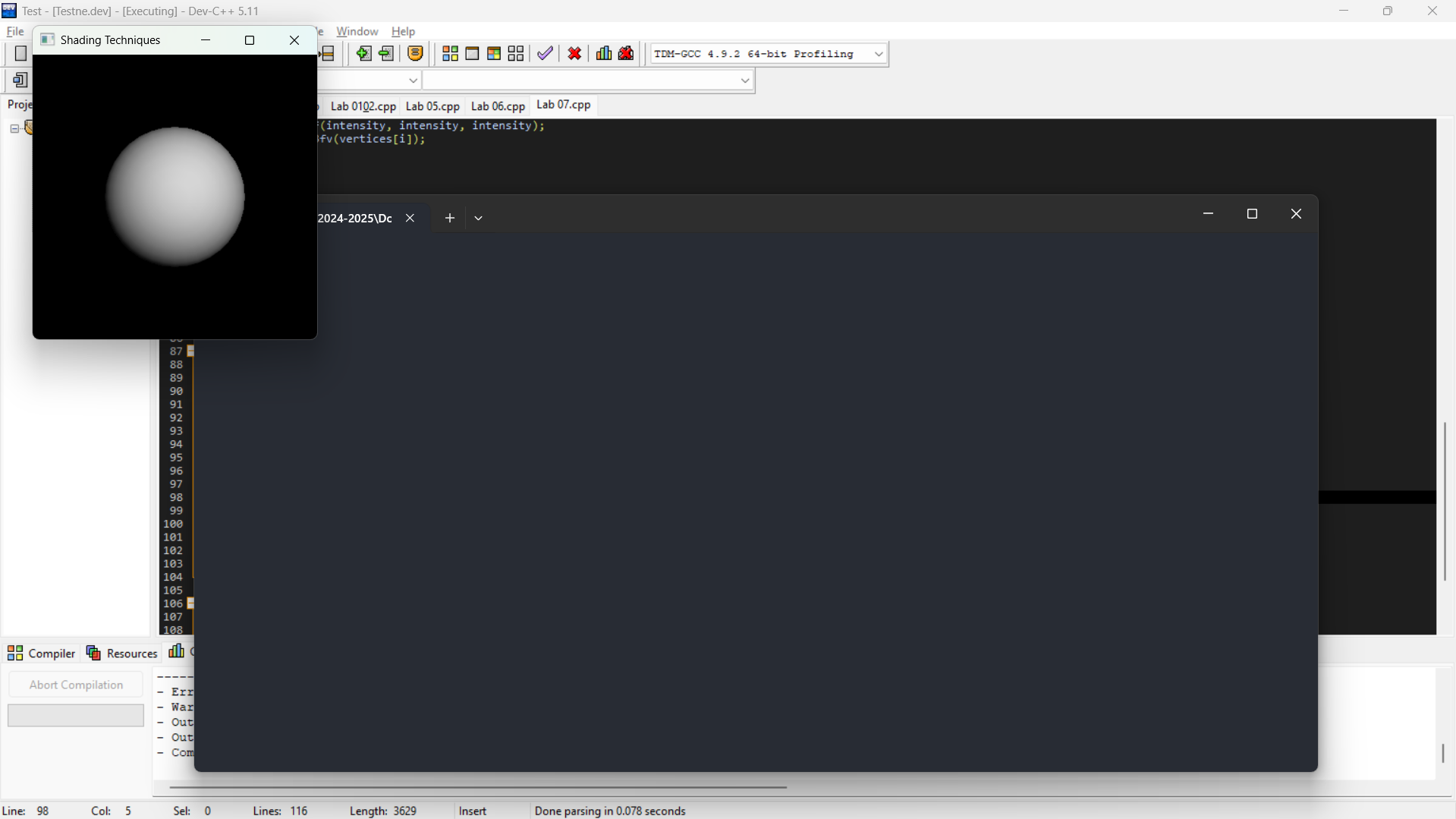
initLighting();

glutDisplayFunc(display);

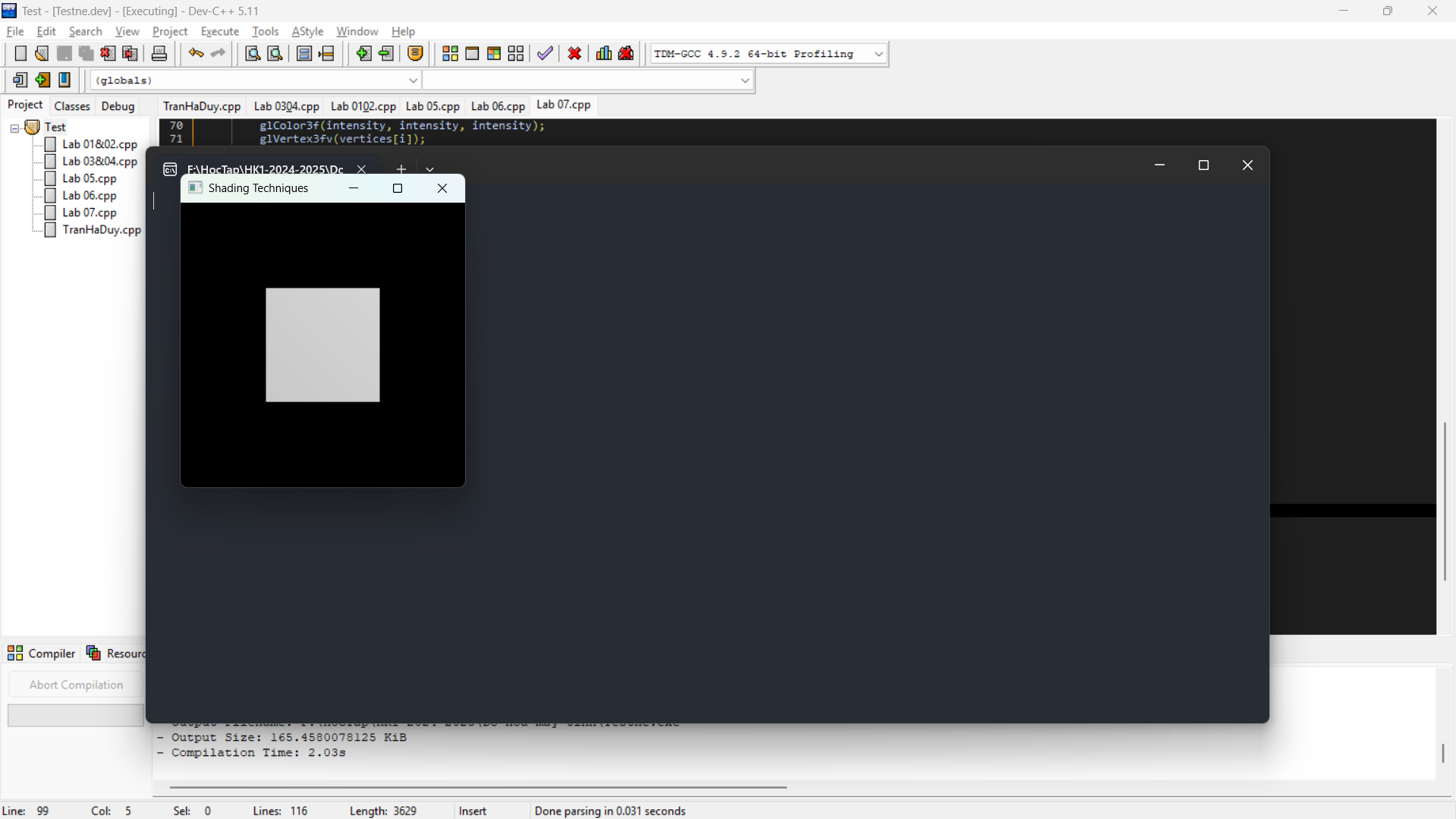
glutMainLoop();

return 0;

}



**Kết quả Tô bóng hằng Lambert**



**Kết quả Tô bóng nội suy Gouraud**

A screenshot of a computer

Description automatically generated

**Kết quả Tô bóng nội suy Phong**