Alexandria University

Faculty of Engineering

Computer Graphics

Lab #2 OpenGL

Name: Toka Ashraf Abo Elwafa Ahmed

ID: 19015539

Problem Statement:

You are required to create an OpenGL project using the project template. You should implement an application that asks user about the projection type (orthographic or perspective). If user chooses orthographic projection a triangle should be drawn; otherwise, a pyramid should be drawn.

At runtime, Input handling should be as follows:

• When user clicks left mouse button, scene should spin around specific axis in counter-clockwise(CCW) manner.

• When user clicks right mouse button, scene should spin around specific axis in clockwise(CW) manner.

• When user presses space button, scene should stop spinning.

• When user presses 'i', you should zoom in the scene.

• When user presses 'o', you should zoom out the scene.

If user chooses parallel projection, the spinning should be around z-axis; otherwise spinning should be around y axis.

Flow Of the Code:

1- Function of drawScene():

- In case of orthographic projection (ortho):
  - spinning is handled using glRotatef(spinZ, 0, 0, 1);
  as it makes spinning around z axis using spinZ that will be updated in the function of spinDisplay() and spinDisplayReverse().
  - Drawing triangle with mentioned dimensions

```
case ortho:
        glRotatef( angle: spinZ,  x: 0,  y: 0,  z: 1);
        glBegin( mode: GL_TRIANGLES);
        glVertex3f( x: 0,  y: 10,  z: 0);
        glVertex3f( x: -30,  y: 0,  z: 0);
        glVertex3f( x: 30,  y: 0,  z: 0);
        glEnd();

    break;
```

- In case of perspective:
  - Spanning is handled using glRotatef(spinY, 0, 1, 0); as it makes spinning around y axis using spinY that will be updated in the function of spinDisplay() and spinDisplayReverse().
  - Zoom in/out are handled using glTranslatef(-5, 0, zOffset*2); and variable of zOffset will be updated in function of keyInput() in case of zoom in and zoom out.
  - Drawing pyramid as four triangles

```
case prespective:
        glRotatef( angle: spinY,  x: 0,  y: 1,  z: 0);
        glTranslatef(  x: -5,  y: 0,  z: zOffset*2);
        glBegin( mode: GL_TRIANGLES);
        glVertex3f(  x: 0,  y: 5,  z: 0);
        glVertex3f( x: -5,  y: -5,  z: 5);
        glVertex3f( x: 5,  y: -5,  z: 5);

        glVertex3f( x: 0,  y: 5,  z: 0);
        glVertex3f( x: 5,  y: -5,  z: 5);
        glVertex3f( x: 5,  y: -5,  z: -5);

        glVertex3f( x: 0,  y: 5,  z: 0);
        glVertex3f( x: 5,  y: -5,  z: -5);
        glVertex3f( x: -5,  y: -5,  z: -5);

        glVertex3f(  x: 0,  y: 5,  z: 0);
        glVertex3f( x: -5,  y: -5,  z: -5);
        glVertex3f( x: -5,  y: -5,  z: 5);
        glEnd();
    break;
```

2- Function of spinDisplay():

It is responsible for spinning in counterclockwise manner -

- by updating the variable of currentSpin:

Using expression of

currentSpin += spinSpeed *(glutGet(GLUT_ELAPSED_TIME) - prevTime) / 100;

multiplying the spinSpeed variable by some ration to decrease the rate of rotation. This ratio subtracts the previous time from the delta time that obtained from function of glutGet(GLUT_ELAPSED_TIME) and divides them by 100 then adds this expression to the current spin.

- Setting the previous time to the delta time.
- At the end, set the current spin to spinZ in case of ortho and set it to spinY in case of perspective.
- Then call function of drawScene() to draw the new shape and call function of glutPostRedisplay() to rerender the marked window.

```
void spinDisplay(void)
{
    glMatrixMode( mode: GL_PROJECTION);
    currentSpin += spinSpeed * (glutGet( query: GLUT_ELAPSED_TIME) - prevTime) / 100;
    prevTime = glutGet( query: GLUT_ELAPSED_TIME);
    switch (userChoice) {
        case ortho :
            spinZ=currentSpin;
            break;
        case prespective:
            spinY=currentSpin;
            break;
        default:
            break;

    }
    glMatrixMode( mode: GL_MODELVIEW);
    drawScene();
    glutPostRedisplay();
```

3- Function of spinDisplayReverse():

It is responsible to make spinning in clockwise manner:

It is implemented as the function of spinDisplay() but subtracting the expression from the current spin to rotate in the opposite direction.

```
void spinDisplayReverse(void)
{
    glMatrixMode( mode: GL_PROJECTION);
    currentSpin -= spinSpeed * (glutGet( query: GLUT_ELAPSED_TIME) - prevTime) / 100;
    prevTime = glutGet( query: GLUT_ELAPSED_TIME);
    switch (userChoice) {
        case ortho :
            spinZ=currentSpin;
            break;
        case prespective:
            spinY=currentSpin;
            break;
        default:
            break;
    }
    glMatrixMode( mode: GL_MODELVIEW);
    drawScene();
    glutPostRedisplay();
```

4- Function of mouse():

- In case of CCW: a call to function of spinDisplay is made using glutIdleFunc.

```
case CCW:
    glutIdleFunc( callback: spinDisplay);
```

- In case of CW: a call to function of spinDisplayReverse is made using glutIdleFunc.

```
case CW:
    glutIdleFunc( callback: spinDisplayReverse);
```

5- Function of keyInput():

- In case of zoom in: the variable of zOffset is increased by 0.1 then call function of drawScene to draw the new shape after zooming in. at the end, call function of glutPostRedisplay() to rerender the marked window.

```
case zoomIn:
        zOffset+=0.1;
        drawScene();
        glutPostRedisplay();
```

- In case of zoom out: the variable of zOffset is decreased by 0.1 then call function of drawScene to draw the new shape after zooming out. at the end, call function of glutPostRedisplay() to rerender the marked window.

```
case zoomOut:
      zOffset-=0.1;
        drawScene();
        glutPostRedisplay();
```

- In case of stopSpinning: a call to NULL is made using glutIdleFunc to stop the rotation.

```
case stopSpinning:
    glutIdleFunc( callback: NULL);
```

6- Function of resize():

- In the case of ortho, initialization of viewing box for parallel projection using ortho variables is made.

```
case ortho:
    glOrtho(orthoLeft , orthoRight, orthoBottom , orthoTop , orthoNear , orthoFar);
```

- In case of perspective, initialization of frustum for perspective projection using fru variables.

```
case prespective:
    glFrustum( left: fruLeft , right: fruRight , bottom: fruBottom , top: fruTop , zNear: fruNear , zFar: fruFar);
```

7- Function of printUserInteraction():

Some statements are printed to show to the user how to deal with the program.

```
void printUserInteraction() {
    std::cout << "click left mouse to spin around specific axis in counter-clockwise(CCW) \n"
                 "click right mouse to spin around specific axis in clockwise(CCW) \n"
                 "press space button to stop spinning \n"
                 "press i button to zoom in \n"
                 "press o button to zoom out\n";
    /*write code below:
     1- print user interaction(good practice)
    */
}
```

8- Main Function:

The input from user is taken as shown:

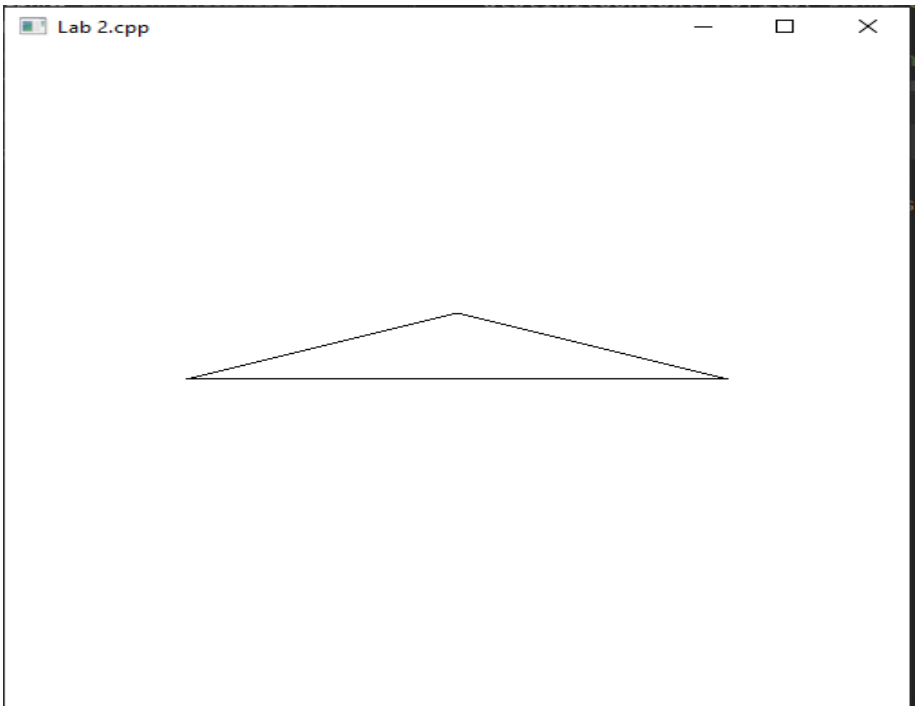
```
std::cin >> userChoice;
```

Test Cases:

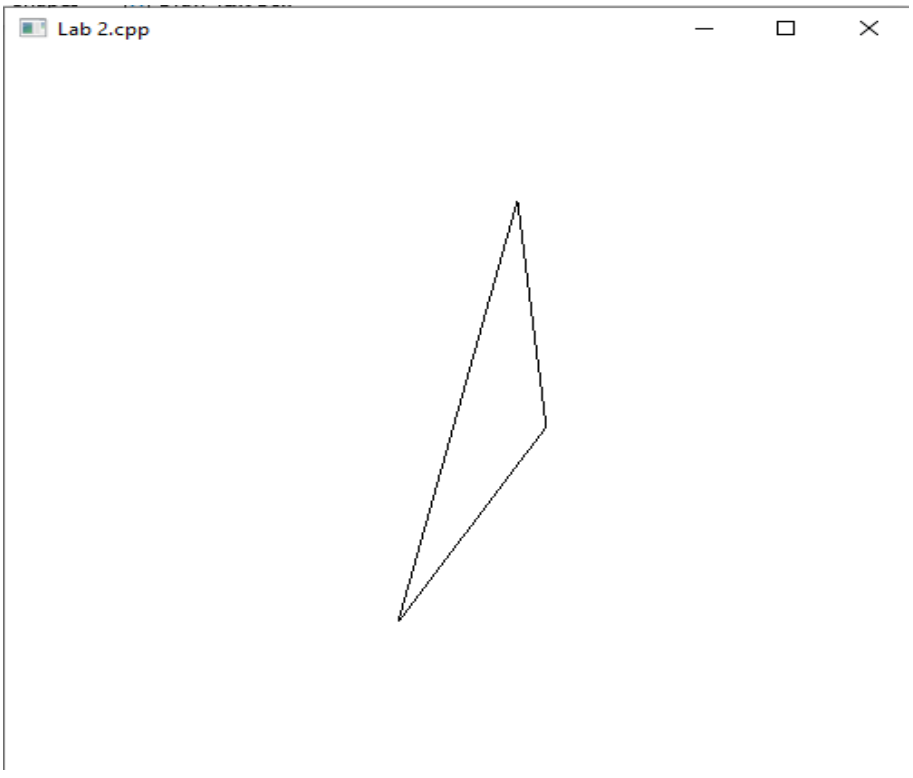
After running the program, some statements will be printed to choose parallel projection, enter 1 and to choose perspective projection, enter 2.

```
E:\CLionProjects\lab_2_graphics\cmake-build-debug\lab_2_graphics.exe
click left mouse to spin around specific axis in counter-clockwise(CCW)
click right mouse to spin around specific axis in clockwise(CCW)
press space button to stop spinning
press i button to zoom in
press o button to zoom out
Which projection type do you want
1) parallel projection
2) perspective projection
>>
```

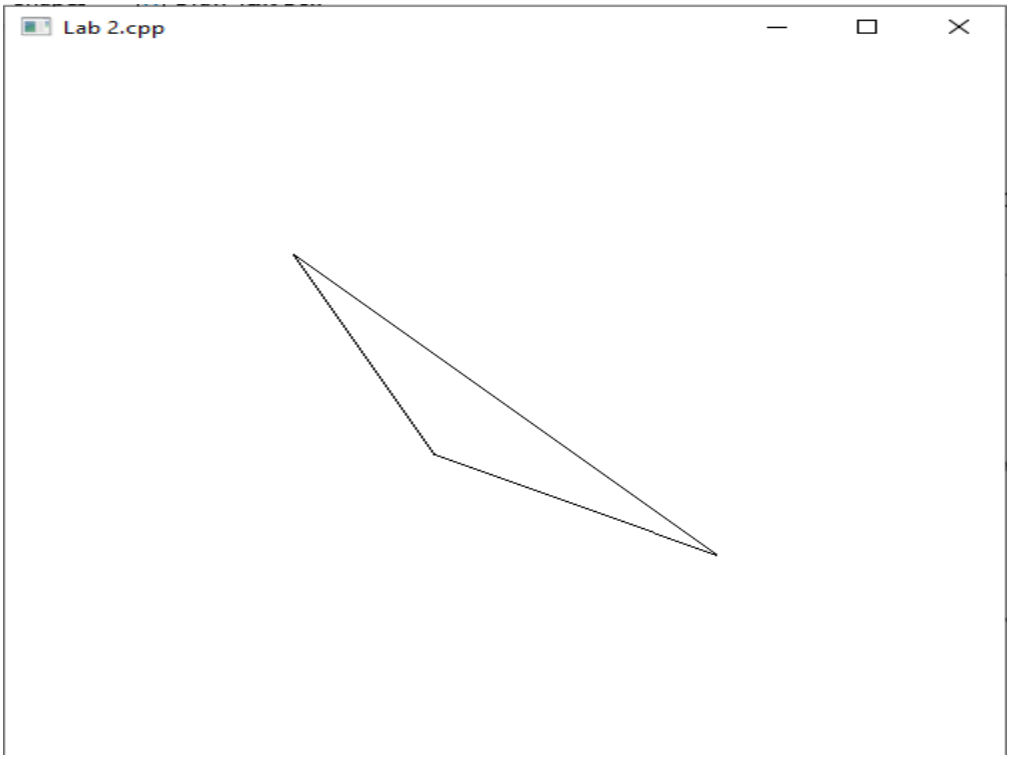
- After choosing 1, the triangle will be drawn:



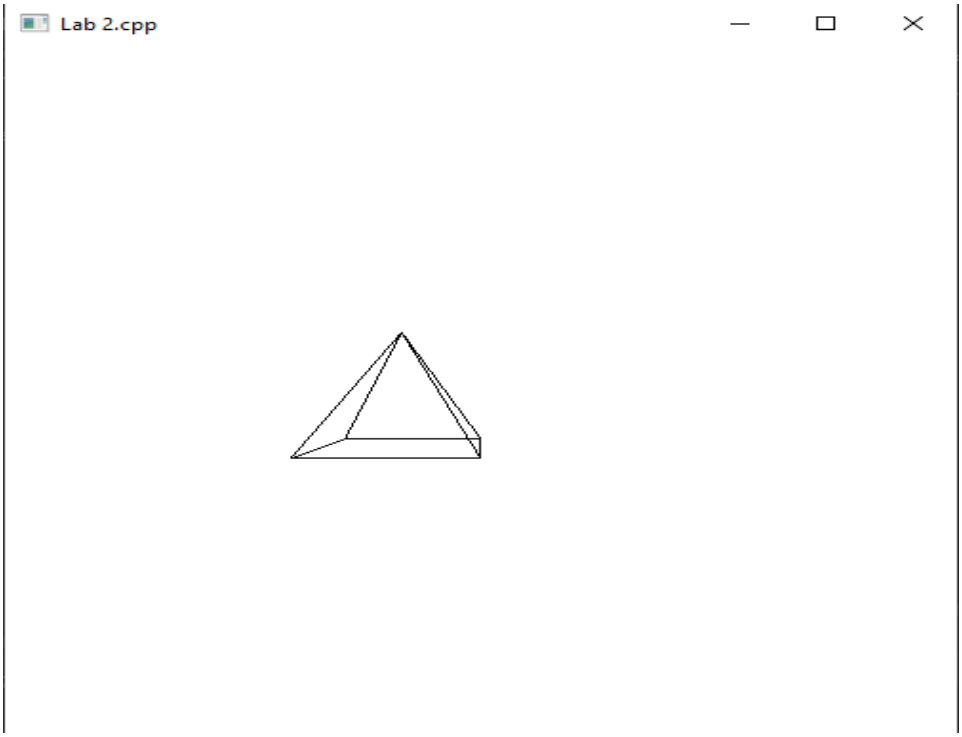
After clicking on left button of the mouse, it makes rotate in counterclockwise manner:



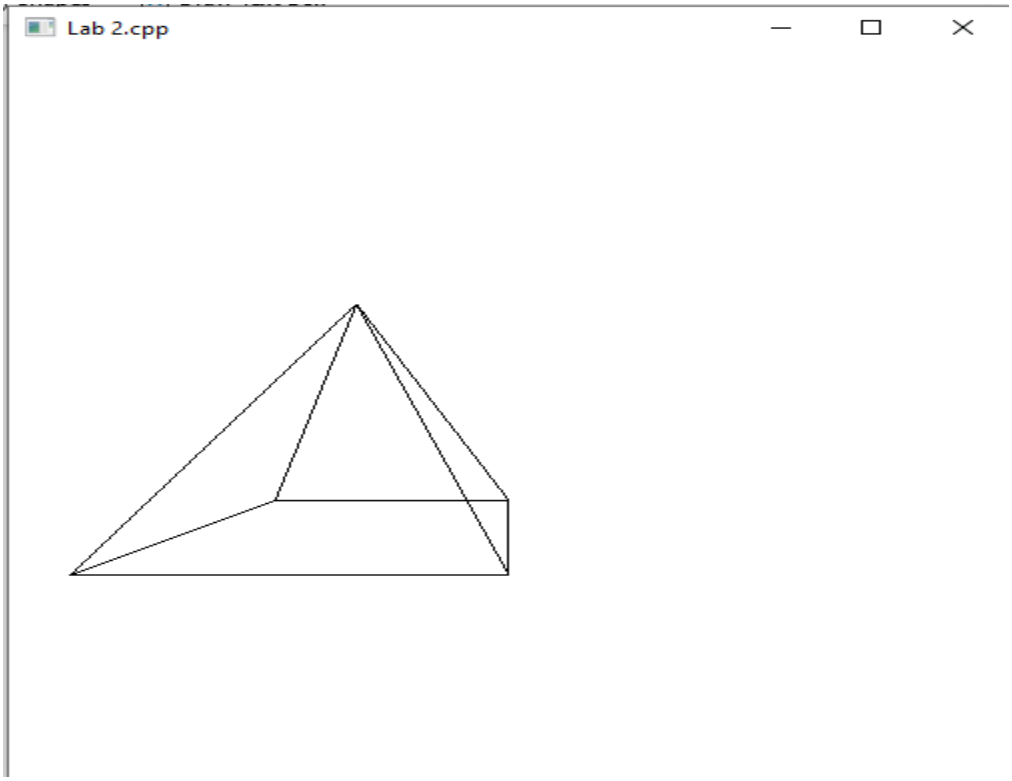
After clicking on right button of the mouse, it makes rotate in clockwise manner:



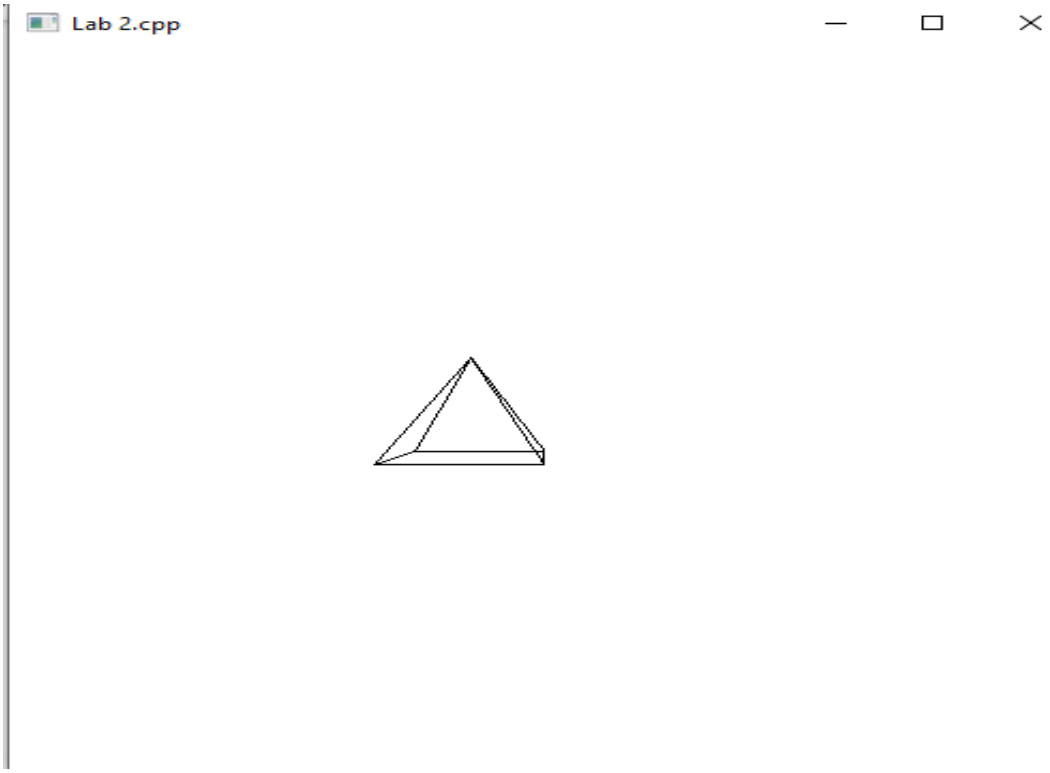
- After choosing 2, the pyramid will be drawn:



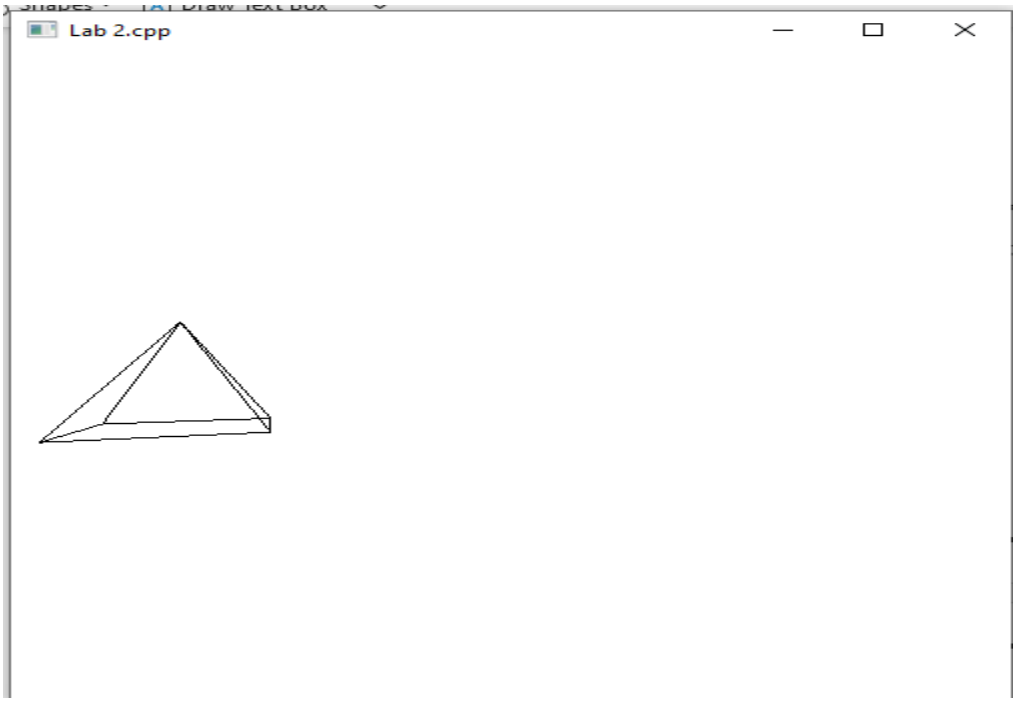
After pressing button of “I” to zoom in:



After pressing button of “o” to zoom out:



After clicking on left button of the mouse to rotate in CCW:



After clicking on right button of the mouse to rotate in CW: