Graphics

Lab 2

Name / Ahmed Abdallah Aboeleid

ID / 19015274

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

Code:

```
switch (userChoice) {
        case ortho:{
    glRotatef(currentSpin, 0.0, 0.0, 1.0);
    glBegin(GL_TRIANGLES);
    glVertex3f(0.0, 10.0, 0.0);
    glVertex3f(-30.0, 0.0, 0.0);
    glVertex3f(30.0, 0.0, 0.0);
    glEnd();
    spinY = 1.0;}
                break;
        case prespective:{
          glTranslatef(0.0, 0.0, zOffset);
    glRotatef(currentSpin, 0.0, 1.0, 0.0);
    glBegin(GL_TRIANGLES);
    glVertex3f(-5.0f, -5.0f, 5.0f);
    glVertex3f(5.0f, -5.0f, 5.0f);
    glVertex3f(0.0f, -5.0f, -5.0f);
    glVertex3f(-5.0f, -5.0f, 5.0f);
    glVertex3f(5.0f, -5.0f, 5.0f);
    glVertex3f(0.0f, 5.0f, 0.0f);
    glVertex3f(5.0f, -5.0f, 5.0f);
    glVertex3f(0.0f, -5.0f, -5.0f);
    glVertex3f(0.0f, 5.0f, 0.0f);
    glVertex3f(-5.0f, -5.0f, 5.0f);
    glVertex3f(0.0f, -5.0f, -5.0f);
    glVertex3f(0.0f, 5.0f, 0.0f);
    glEnd();}
                break;
        default:
                break;
        }
```

In this part of code in drawscene function ,I define part which responsible for drawing triangle and code for drawing pyramids.

```
void spinDisplay(void)
{
   float currentTime = glutGet(GLUT_ELAPSED_TIME);
   float deltaTime = currentTime - prevTime;
   currentSpin += spinSpeed * deltaTime / 1000.0;
   glutPostRedisplay();
   prevTime = currentTime;
}
```

In this function, we get elapsed time in msec since the start of the GLUT program.

And get delta time to use it in calculating angle which object will spin by it.

In clockwise.

```
void spinDisplayReverse(void)
{
  float currentTime = glutGet(GLUT_ELAPSED_TIME);
   float deltaTime = currentTime - prevTime;
   currentSpin -= spinSpeed * deltaTime / 1000.0;
   glutPostRedisplay();
   prevTime = currentTime;
}
```

In this function, we get elapsed time in msec since the start of the GLUT program.

And get delta time to use it in calculating angle which object will spin by it.

In counter clockwise.

```
void mouse(int button, int state, int x, int y)
{   switch (button) {
            case CCW:
       if (state == GLUT_DOWN){
          glutIdleFunc(spinDisplay);}
        break;
            case CW:
                        if (state == GLUT_DOWN){
                          glutIdleFunc(spinDisplayReverse); }
            }
}
```

In this function we take input from mouse , and check if button of click is pressed or not and if user press right or left click to determine direction of spinning

```
void keyInput(unsigned char key, int x, int y)
{
	switch (key)
	{
	case 27:
		exit(0);
		break;
	case zoomIn:
		zOffset += 1.0;
		break;
	case zoomOut:
		zOffset -= 1.0;
		break;
	case stopSpinning:
		glutIdleFunc(NULL);
		break;
	default:
		break;
	}
	glutPostRedisplay();
}
```

In this function, I take input from user when press button in keyboard and determine action according to button which user pressed ,

If press I => zoom in

If press o => zoom out

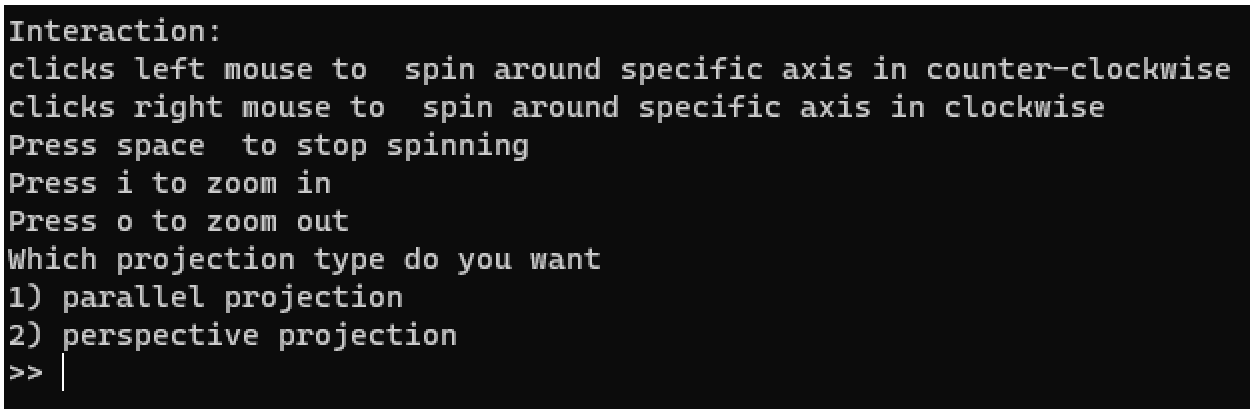
If press space => stop spinning

```
void resize(int w, int h)

{

        glViewport(0, 0, w, h);

        glMatrixMode(GL_PROJECTION);

        glLoadIdentity();

        switch (userChoice) {

        case ortho:

                glOrtho(orthoLeft, orthoRight, orthoBottom, orthoTop, orthoNear, orthoFar);

                spinY = 1.0;

                break;

        case prespective:

                glFrustum(fruLeft, fruRight, fruBottom, fruTop, fruNear, fruFar);

                spinZ = 1.0;

                break;

        default:

                break;

        }

        glMatrixMode(GL_MODELVIEW);

}
```

In function resize , it used to redraw scene with new dimensions according to new window width and height after user resizing window

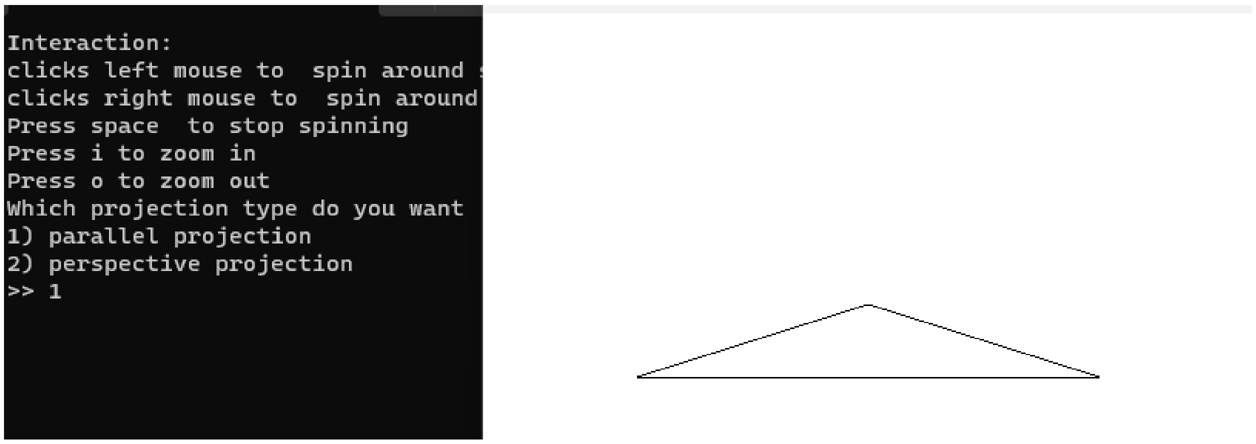
Screenshoots:

- ⇨ start screen



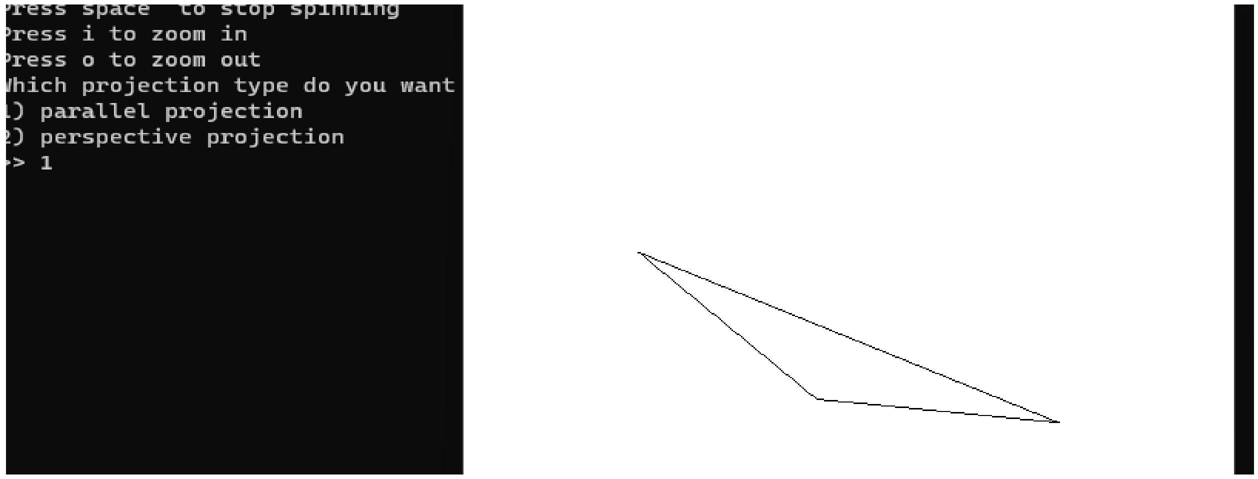
```
Interaction:
clicks left mouse to  spin around specific axis in counter-clockwise
clicks right mouse to  spin around specific axis in clockwise
Press space  to stop spinning
Press i to zoom in
Press o to zoom out
Which projection type do you want
1) parallel projection
2) perspective projection
>> 
```

- ⇨ draw triangles

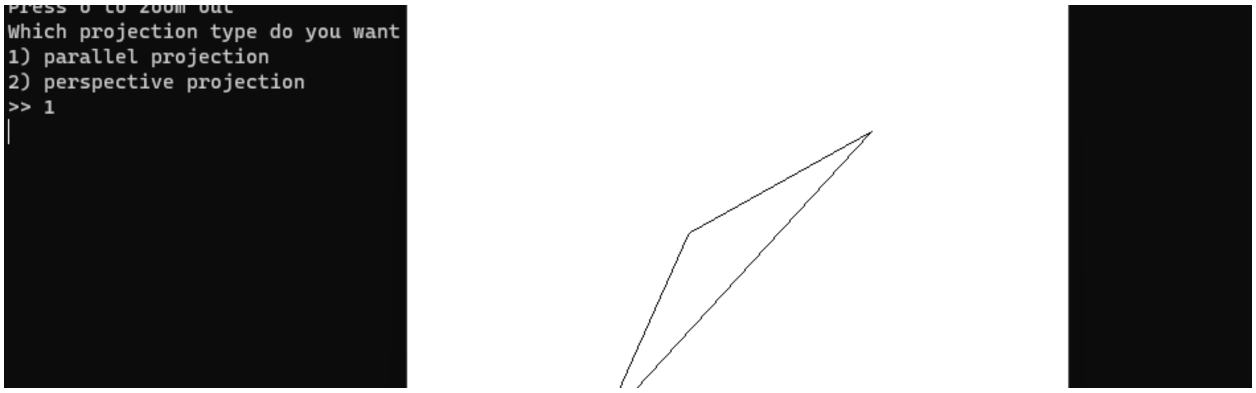


```
Interaction:
clicks left mouse to  spin around s
clicks right mouse to  spin around
Press space  to stop spinning
Press i to zoom in
Press o to zoom out
Which projection type do you want
1) parallel projection
2) perspective projection
>> 1
```

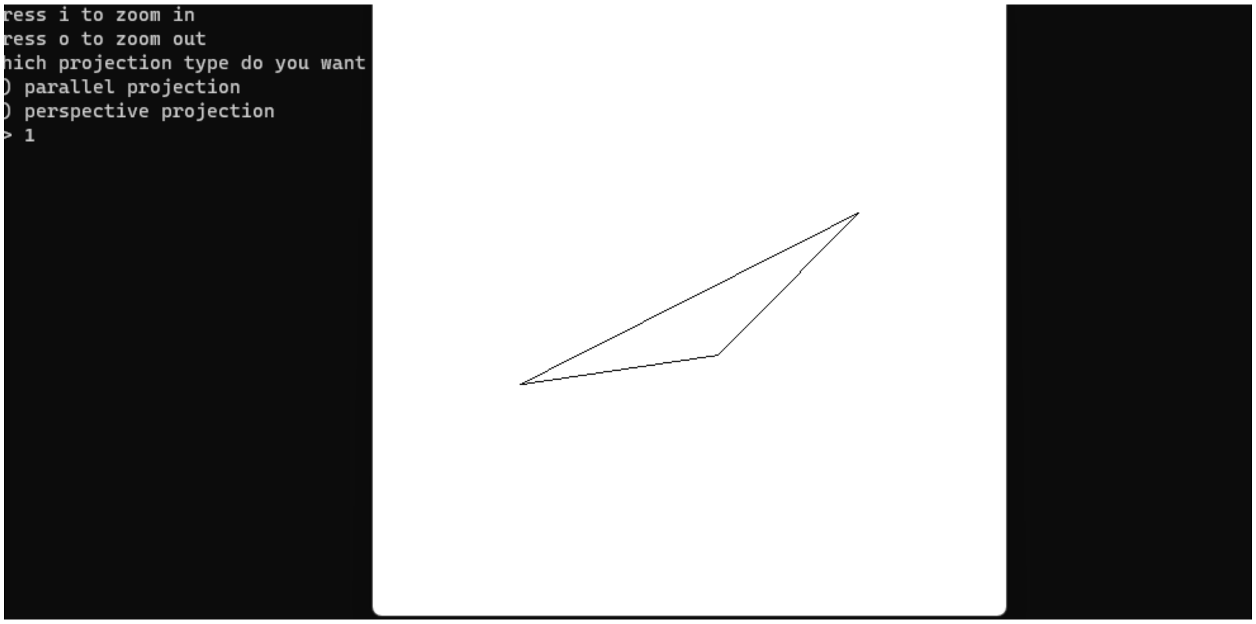
- ⇨ spinning



```
ress space  to stop spinning
ress i to zoom in
ress o to zoom out
hich projection type do you want
) parallel projection
) perspective projection
> 1
```

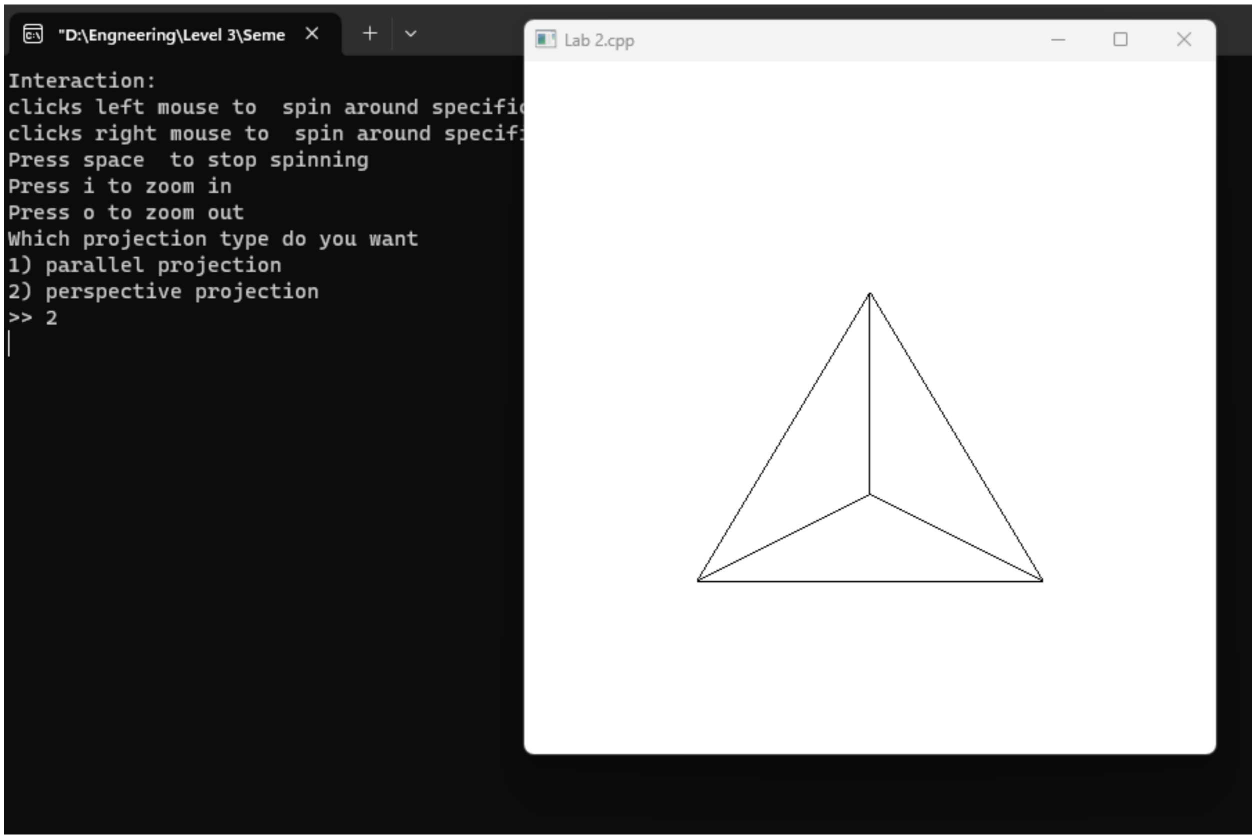


```
Press o to zoom out
Which projection type do you want
1) parallel projection
2) perspective projection
>> 1
```



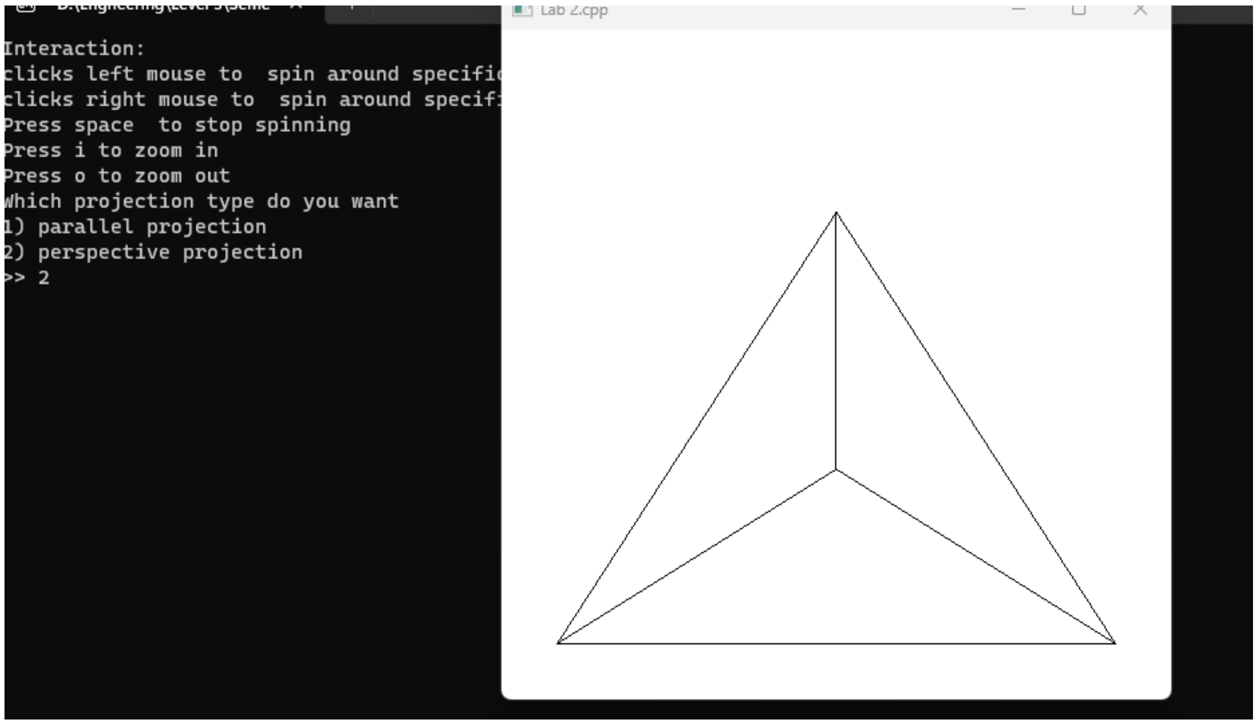
```
ress i to zoom in
ress o to zoom out
hich projection type do you want
) parallel projection
) perspective projection
> 1
```

⇨ Draw Pyramids



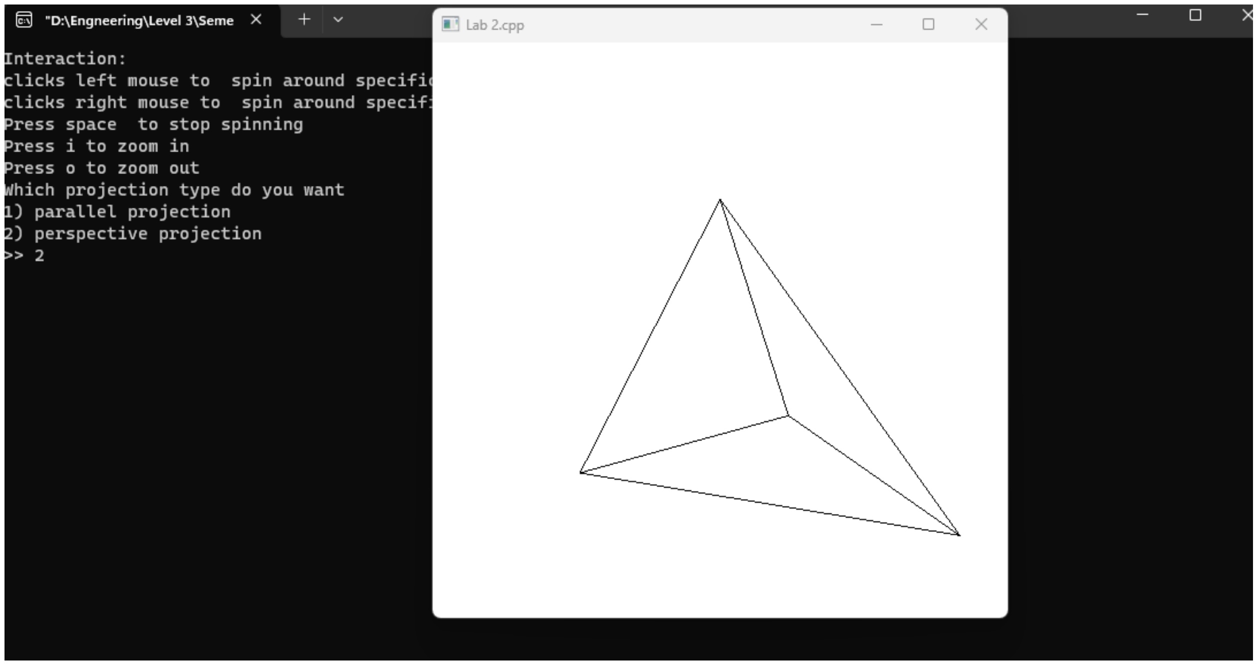
```
"D:\Engneering\Level 3\Seme
Interaction:
clicks left mouse to  spin around specifi
clicks right mouse to  spin around specifi
Press space  to stop spinning
Press i to zoom in
Press o to zoom out
Which projection type do you want
1) parallel projection
2) perspective projection
>> 2
```

⇨ Zoom In and out :

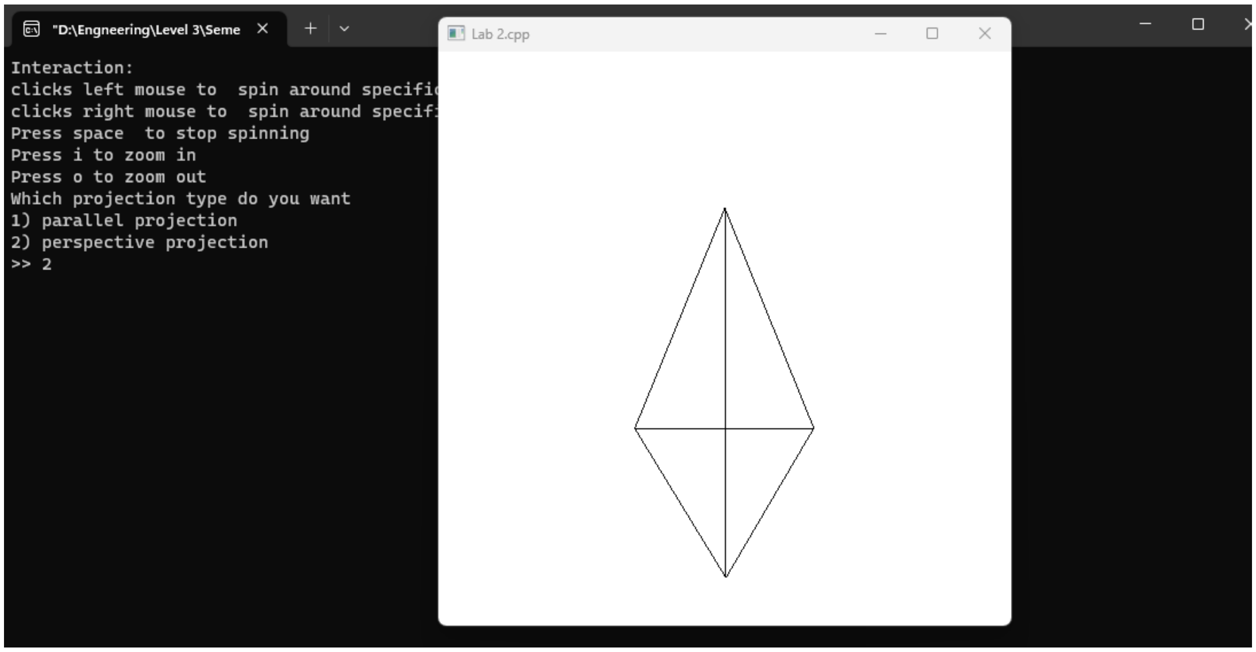


```
Interaction:
clicks left mouse to  spin around specific
clicks right mouse to  spin around specifi
Press space  to stop spinning
Press i to zoom in
Press o to zoom out
Which projection type do you want
1) parallel projection
2) perspective projection
>> 2
```

⇨ Spin

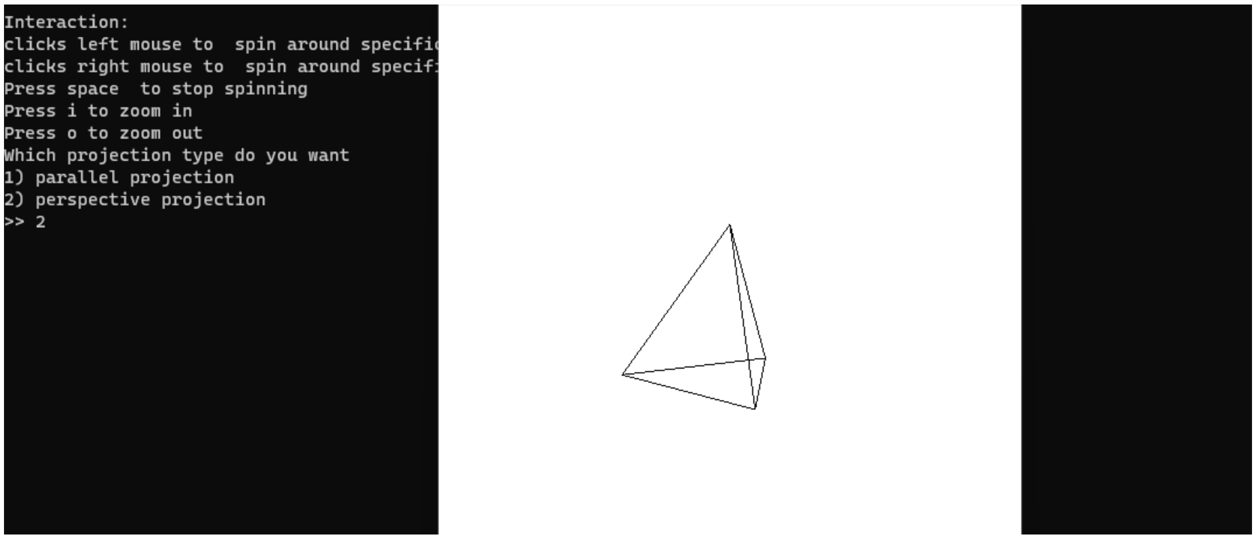


```
Interaction:
clicks left mouse to  spin around specific
clicks right mouse to  spin around specifi
Press space  to stop spinning
Press i to zoom in
Press o to zoom out
Which projection type do you want
1) parallel projection
2) perspective projection
>> 2
```



```
Interaction:
clicks left mouse to  spin around specifi
clicks right mouse to  spin around specif
Press space  to stop spinning
Press i to zoom in
Press o to zoom out
Which projection type do you want
1) parallel projection
2) perspective projection
>> 2
```

⇨ **Spin with zoom in and out**



```
Interaction:
clicks left mouse to  spin around specifi
clicks right mouse to  spin around specif
Press space  to stop spinning
Press i to zoom in
Press o to zoom out
Which projection type do you want
1) parallel projection
2) perspective projection
>> 2
```