UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

LABORATORIO DE COMPUTACIÓN GRÁFICA e INTERACCIÓN HUMANO COMPUTADORA

REPORTE DE PRÁCTICA Nº 03

NOMBRE COMPLETO: Hernández Domínguez Luis Carlos

Nº de Cuenta: 320182668

GRUPO DE LABORATORIO: 03

GRUPO DE TEORÍA: 04

SEMESTRE 2026-1

FECHA DE ENTREGA LÍMITE: 07/09/2025

CALIFICACIÓN: _____

REPORTE DE PRÁCTICA:

1. Ejercicios

Ejercicio: Generar una pirámide rubik (pyraminx) de 9 **pirámides** por cara. Cada cara de la pyraminx que se vea de un color diferente y que se vean las separaciones entre instancias (las líneas oscuras son las que permiten diferenciar cada pirámide pequeña).

Bloques de código generado

```
387         //Piramide negra
388         model = glm::mat4(1.0);
389         model = glm::translate(model, glm::vec3(0.0f, 1.0f, 0.0f));
390         model = glm::rotate(model, glm::radians(mainWindow.getrotax()), glm::vec3(1.0f, 0.0f, 0.0f));
391         model = glm::rotate(model, glm::radians(mainWindow.getrotay()), glm::vec3(0.0f, 1.0f, 0.0f));
392         model = glm::rotate(model, glm::radians(mainWindow.getrotaz()), glm::vec3(0.0f, 0.0f, 1.0f));
393         model = glm::scale(model, glm::vec3(2.1f, 2.1f, 2.1f));
394         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
395         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
396         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
397         color = glm::vec3(0.0f, 0.0f, 0.0f);
398         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
399         meshList[1]->RenderMesh();
400
401
402         //Cara roja
403
404         // Punta
405         model = glm::mat4(1.0);
406         model = glm::translate(model, glm::vec3(0.0f, 1.55f, -0.295f));
407         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
408         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
409         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
410         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
411         color = glm::vec3(1.0f, 0.0f, 0.0f);
412         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
413         meshList[1]->RenderMesh();
```

```
415         //Sección media
416         model = glm::mat4(1.0);
417         model = glm::translate(model, glm::vec3(0.0f, 0.96f, -0.125f));
418         model = glm::rotate(model, 180 * toRadians, glm::vec3(0.0f, 0.0f, 1.0f));
419         model = glm::rotate(model, 30 * toRadians, glm::vec3(1.0f, 0.0f, 0.0f));
420         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
421         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
422         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
423         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
424         color = glm::vec3(1.0f, 0.0f, 0.0f);
425         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
426         meshList[1]->RenderMesh();
427
428         model = glm::mat4(1.0);
429         model = glm::translate(model, glm::vec3(0.33f, 0.92f, -0.135f));
430         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
431         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
432         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
433         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
434         color = glm::vec3(1.0f, 0.0f, 0.0f);
435         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
436         meshList[1]->RenderMesh();
437
438         model = glm::mat4(1.0);
439         model = glm::translate(model, glm::vec3(-0.33f, 0.92f, -0.135f));
440         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
441         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
442         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
443         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
444         color = glm::vec3(1.0f, 0.0f, 0.0f);
445         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
446         meshList[1]->RenderMesh();
```

```
448         //Sección baja
449         model = glm::mat4(1.0);
450         model = glm::translate(model, glm::vec3(0.0f, 0.3f, 0.0f));
451         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
452         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
453         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
454         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
455         color = glm::vec3(1.0f, 0.0f, 0.0f);
456         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
457         meshList[1]->RenderMesh();
458
459
460         model = glm::mat4(1.0);
461         model = glm::translate(model, glm::vec3(-0.65f, 0.3f, 0.0f));
462         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
463         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
464         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
465         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
466         color = glm::vec3(1.0f, 0.0f, 0.0f);
467         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
468         meshList[1]->RenderMesh();
469
470         model = glm::mat4(1.0);
471         model = glm::translate(model, glm::vec3(0.65f, 0.3f, 0.0f));
472         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
473         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
474         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
475         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
476         color = glm::vec3(1.0f, 0.0f, 0.0f);
477         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
478         meshList[1]->RenderMesh();
```

```
504         //Cara verde
505
506         // Punta
507         model = glm::mat4(1.0);
508         model = glm::translate(model, glm::vec3(0.05f, 1.55f, -0.4f));
509         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
510         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
511         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
512         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
513         color = glm::vec3(0.0f, 0.8f, 0.0f);
514         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
515         meshList[1]->RenderMesh();
516
517         //Sección media
518         //(0.5f,-0.5f,0.0f)   1
519         //0.0f, 0.5f, -0.25f  2
520         //0.0f, -0.5f, -0.5f  3
521         // 1 - 2 = (0.5,-0.5,0.25) = â
522         // 3 - 2 = (0,-1,-0.25) = â2
523         // âxâ2= (0.6963106238, 0.174077656, -0.6963106238)
524         model = glm::mat4(1.0);//0.07
525         model = glm::translate(model, glm::vec3(0.19f, 0.92f, -0.52f));
526         model = glm::scale(model, glm::vec3(0.49f, 0.49f, 0.49f));
527         model = glm::rotate(model, 180 * toRadians, glm::vec3(0.6963106238f, 0.174077656f, -0.6963106238f));
528         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
529         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
530         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
531         color = glm::vec3(0.0f, 0.8f, 0.0f);
532         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
533         meshList[1]->RenderMesh();
```

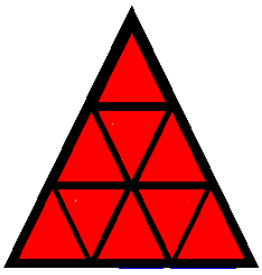
```
535         model = glm::mat4(1.0);
536         model = glm::translate(model, glm::vec3(0.05f, 0.96f, -0.56f));
537         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
538         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
539         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
540         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
541         color = glm::vec3(0.0f, 0.8f, 0.0f);
542         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
543         meshList[1]->RenderMesh();
544
545         model = glm::mat4(1.0);
546         model = glm::translate(model, glm::vec3(0.35f, 0.96f, -0.25f));
547         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
548         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
549         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
550         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
551         color = glm::vec3(0.0f, 0.8f, 0.0f);
552         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
553         meshList[1]->RenderMesh();
554
555         //Sección baja
556
557         model = glm::mat4(1.0);
558         model = glm::translate(model, glm::vec3(0.05f, 0.34f, -0.7f));
559         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
560         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
561         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
562         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
563         color = glm::vec3(0.0f, 0.8f, 0.0f);
564         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
565         meshList[1]->RenderMesh();
```

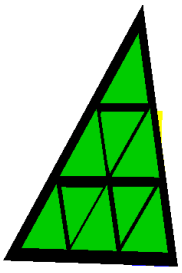
```
609         //Cara amarilla
610
611         //Punta
612         model = glm::mat4(1.0);
613         model = glm::translate(model, glm::vec3(-0.05f, 1.55f, -0.4f));
614         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
615         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
616         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
617         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
618         color = glm::vec3(1.0f, 1.0f, 0.0f);
619         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
620         meshList[1]->RenderMesh();
621
622         //Sección media
623         //(-0.5f,-0.5f,0.0f)      1
624         //0.0f, 0.5f, -0.25f  2
625         //0.0f, -0.5f, -0.5f  3
626         // 1 - 2 = (-0.5,-1,0.25) = â
627         // 3 - 2 = (0,-1,-0.25) = â2
628         // (0.5,-0.125,0.5)
629         // âxâ2= (0.6963106238, -0.174077656, 0.6963106238)
630         model = glm::mat4(1.0);//0.07
631         model = glm::translate(model, glm::vec3(-0.2f, 0.93f, -0.52f));
632         model = glm::scale(model, glm::vec3(0.49f, 0.49f, 0.49f));
633         model = glm::rotate(model, 180 * toRadians, glm::vec3(0.6963106238f, -0.174077656f, 0.6963106238f));
634         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
635         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
636         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
637         color = glm::vec3(1.0f, 1.0f, 0.0f);
638         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
639         meshList[1]->RenderMesh();
```

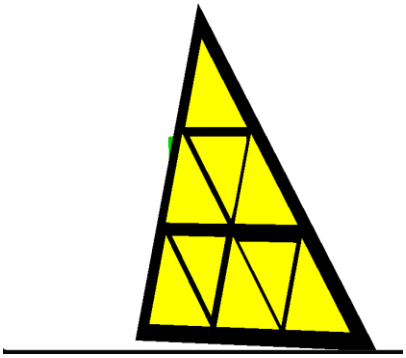
```
716         //Cara azul
717
718         //Punta
719         model = glm::mat4(1.0);
720         model = glm::translate(model, glm::vec3(0.0f, 0.2f, -0.7f));
721         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
722         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
723         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
724         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
725         color = glm::vec3(0.0f, 0.0f, 1.0f);
726         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
727         meshList[1]->RenderMesh();
728
729         //Sección media
730         model = glm::mat4(1.0);
731         model = glm::translate(model, glm::vec3(0.0f, 0.2f, -0.67f));
732         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
733         model = glm::rotate(model, 180 * toRadians, glm::vec3(0.0f, 1.0f, 0.0f));
734         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
735         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
736         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
737         color = glm::vec3(0.0f, 0.0f, 1.0f);
738         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
739         meshList[1]->RenderMesh();
740
741         model = glm::mat4(1.0);
742         model = glm::translate(model, glm::vec3(0.3f, 0.2f, -0.38f));
743         model = glm::scale(model, glm::vec3(0.55f, 0.55f, 0.55f));
744         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
745         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
746         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
747         color = glm::vec3(0.0f, 0.0f, 1.0f);
748         glUniform3fv(uniformColor, 1, glm::value_ptr(color));
749         meshList[1]->RenderMesh();
```

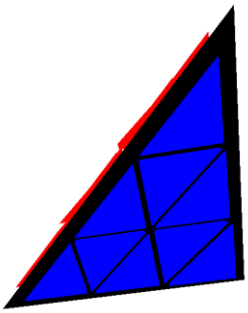
Ejecución

Caras

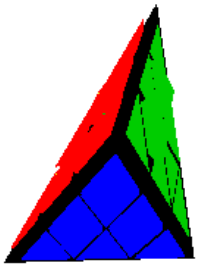








Vistas parciales



Para abordar este problema, en primera instancia se insertó una pirámide principal que sirve como base en la cual insertar las otras pirámides que forman las distintas caras de la figura. La cara azul y roja no fueron ningún problema debido a que la normal de esas caras coincidían con el eje y y z respectivamente.

En cuanto a la cara verde y amarilla se refiere, las pirámides posicionadas como triángulos invertidos fueron la principal problemática. Debido a que, para rotar correctamente estas pirámides era necesario calcular un vector que permitiera rotar entorno a estas caras por lo que se realizó el siguiente procedimiento.

Cara amarilla

$$A = (-0.5, -0.5, 0)$$

$$B = (0, 0.5, -0.25)$$

$$C = (0, -0.5, -0.5)$$

$$A - B = D = (-0.5, -1, 0.25)$$

$$C - B = E = (0, -1, -0.25)$$

Realizando el producto cruz y volviendo el vector unitario

$$\hat{u} = (0.6963106238, -0.174077656, 0.6963106238)$$

Cara verde

$$A = (0.5\ , -0.5\ ,0\ )$$

$$B = (0,0.5, -0.25)$$

$$C = (0, -0.5, -0.5)$$

$$A - B = D = (0.5, -0,5, 0.25)$$

$$C - B = E = (0, -1, -0.25)$$

Realizando el producto cruz y volviendo el vector unitario

$$\hat{u} = (0.6963106238, 0.174077656, -\ 0.6963106238)$$

Una vez conseguidos estos vectores solo fue necesario reemplazarlos en el comando rotate y rotar 180°.

2. Problemas presentados.

No se presentó ningún problema a la hora de ejecutar código.

3.- Conclusión:

a. En esta ocasión, el único ejercicio planteado resultó tener más consideraciones de las cuales yo pensaba inicialmente. El uso del producto cruz para la resolución no fue algo que me pasó por la cabeza inicialmente tras leer la problemática, pero fue algo bastante interesante que me ayudó a comprender de mejor manera este espacio tridimensional.
b. Los conceptos y explicaciones dadas en clase fueron correctas y suficientes para la resolución de esta práctica.
c. En conclusión, esta práctica me pareció tener un nivel de dificultad adecuado y hace un buen trabajo al emplear el dibujo de figuras geométricas de una manera distinta a la práctica anterior.