NOE FRANKLIN CHOQUENAIRA QUISPE 133962

3. ACTIVIDADES EN CLASE

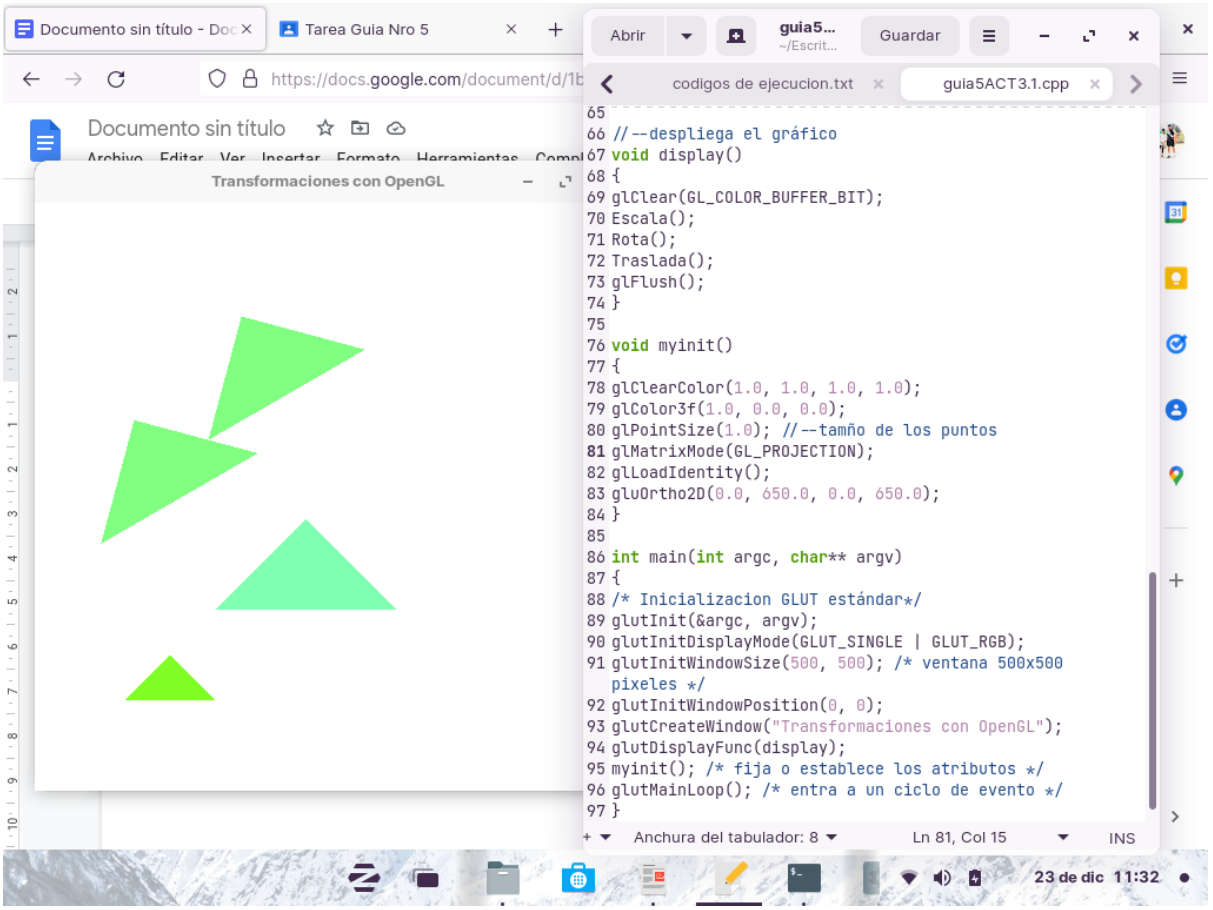
3.1. Escribir un programa que escale por 2 en ambas direcciones, luego rote 30° y finalmente traslade un triángulo en el vector (80,20).

CÓDIGO:

```
#include<math.h>
#include<cstdlib>
#include<GL/glew.h>
#include<GL/glut.h>
#include<stdio.h>
using namespace std;

void Escala()
{
//--triangulo antes de escalar
glColor3f(0.5f, 1.0f, 0.15f);
glBegin(GL_TRIANGLES);
glVertex2f(100.0f, 100.0f);
glVertex2f(200.0f, 100.0f);
glVertex2f(150.0f, 150.0f);
glEnd();
//--triangulo despues de escalar
glScalef(2.0, 2.0, 2.0);
glBegin(GL_TRIANGLES);
glVertex2f(100.0f, 100.0f);
glVertex2f(200.0f, 100.0f);
glVertex2f(150.0f, 150.0f);
glEnd();
}

//rota 30°
void Rota()
{
//--triangulo antes de rotar
glColor3f(0.5f, 1.0f, 0.7f);
glBegin(GL_TRIANGLES);
glVertex2f(100.0f, 100.0f);
glVertex2f(200.0f, 100.0f);
glVertex2f(150.0f, 150.0f);
glEnd();
//--triangulo rotado
glRotatef(30, 0, 0, 1);
glBegin(GL_TRIANGLES);
glVertex2f(100.0f, 100.0f);
glVertex2f(200.0f, 100.0f);
glVertex2f(150.0f, 150.0f);
glEnd();
```

```
}

//---trasladar el triangulo al vector (80,20)
void Traslada()
{
//--triangulo antes de trasladar
glColor3f(0.5f, 1.0f, 0.5f);
glBegin(GL_TRIANGLES);
glVertex2f(100.0f, 100.0f);
glVertex2f(200.0f, 100.0f);
glVertex2f(150.0f, 150.0f);
glEnd();

//--triangulo después de trasladar
glTranslatef(80.0f, 20.0f, 0.0f);
glBegin(GL_TRIANGLES);
glVertex3f(100.0f, 100.0f, 0.0f);
glVertex3f(200.0f, 100.0f, 0.0f);
glVertex3f(150.0f, 150.0f, 0.0f);
glEnd();
}


//--despliega el gráfico
void display()
{
glClear(GL_COLOR_BUFFER_BIT);
Escala();
Rota();
Traslada();
glFlush();
}

void myinit()
{
glClearColor(1.0, 1.0, 1.0, 1.0);
glColor3f(1.0, 0.0, 0.0);
glPointSize(1.0); //--tamño de los puntos
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0.0, 650.0, 0.0, 650.0);
}

int main(int argc, char** argv)
{
/* Inicializacion GLUT estándar*/
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(500, 500); /* ventana 500x500 pixeles */
glutInitWindowPosition(0, 0);
```

```
glutCreateWindow("Transformaciones con OpenGL");
glutDisplayFunc(display);
myinit(); /* fija o establece los atributos */
glutMainLoop(); /* entra a un ciclo de evento */
}
```



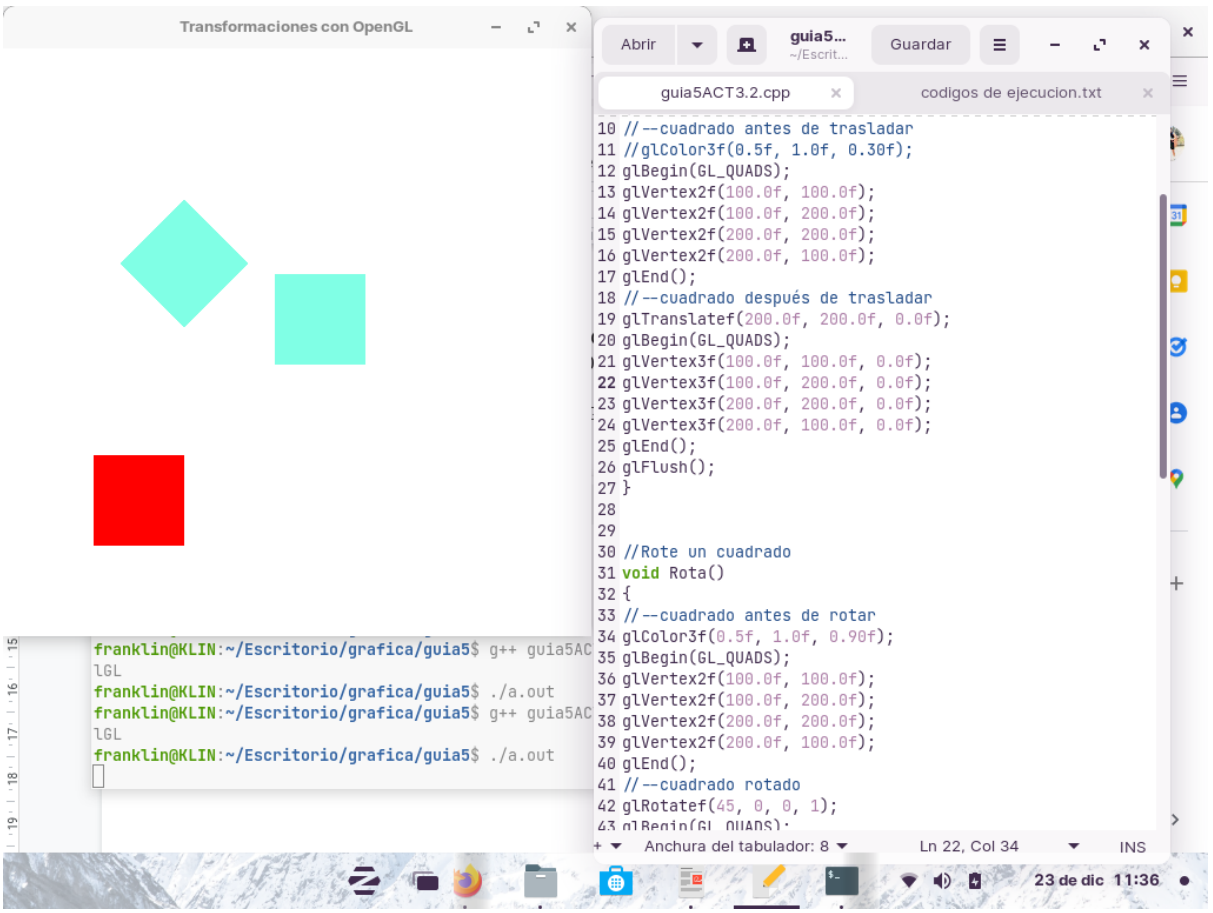
3.2. Escribir un programa que traslade y luego rote un cuadrado: (poner los parámetros de traslación y ángulo de rotación a criterio)

CÓDIGO:

```
#include<math.h>
#include<cstdlib>
#include<GL/glew.h>
#include<GL/glut.h>
#include<stdio.h>
using namespace std;

void Traslada()
{
//--cuadrado antes de trasladar
//glColor3f(0.5f, 1.0f, 0.30f);
glBegin(GL_QUADS);
```

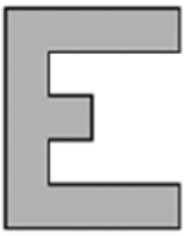
```
glVertex2f(100.0f, 100.0f);
glVertex2f(100.0f, 200.0f);
glVertex2f(200.0f, 200.0f);
glVertex2f(200.0f, 100.0f);
glEnd();
//--cuadrado después de trasladar
glTranslatef(200.0f, 200.0f, 0.0f);
glBegin(GL_QUADS);
glVertex3f(100.0f, 100.0f, 0.0f);
glVertex3f(100.0f, 200.0f, 0.0f);
glVertex3f(200.0f, 200.0f, 0.0f);
glVertex3f(200.0f, 100.0f, 0.0f);
glEnd();
glFlush();
}

//Rote un cuadrado
void Rota()
{
//--cuadrado antes de rotar
glColor3f(0.5f, 1.0f, 0.90f);
glBegin(GL_QUADS);
glVertex2f(100.0f, 100.0f);
glVertex2f(100.0f, 200.0f);
glVertex2f(200.0f, 200.0f);
glVertex2f(200.0f, 100.0f);
glEnd();
//--cuadrado rotado
glRotatef(45, 0, 0, 1);
glBegin(GL_QUADS);
glVertex3f(100.0f, 100.0f, 0.0f);
glVertex3f(100.0f, 200.0f, 0.0f);
glVertex3f(200.0f, 200.0f, 0.0f);
glVertex3f(200.0f, 100.0f, 0.0f);
glEnd();
}

//--despliega el gráfico
void display()
{
glClear(GL_COLOR_BUFFER_BIT);
Traslada();
Rota();
glFlush();
```

```
}

void myinit()
{
glClearColor(1.0, 1.0, 1.0, 1.0);
glColor3f(1.0, 0.0, 0.0);
glPointSize(1.0); //--tamño de los puntos
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0.0, 650.0, 0.0, 650.0);
}

int main(int argc, char** argv)
{
/* Inicializacion GLUT estándar*/
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(500, 500); /* ventana 500x500 pixeles */
glutInitWindowPosition(0, 0);
glutCreateWindow("Transformaciones con OpenGL");
glutDisplayFunc(display);
myinit(); /* fija o establece los atributos */
glutMainLoop(); /* entra a un ciclo de evento */
}
```



4. TAREA.

4.1. Escribir un programa para realizar una animación de rotación de la siguiente figura.



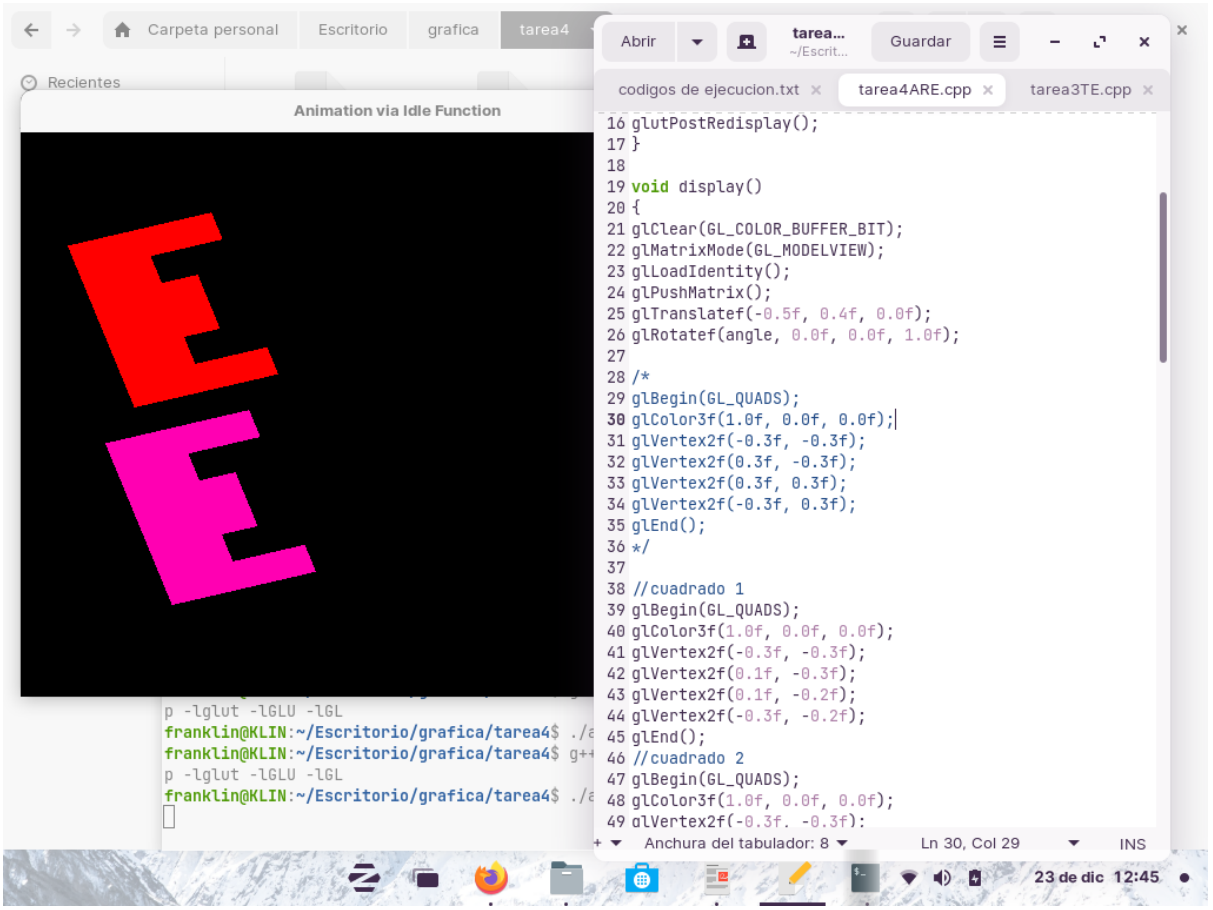
CÓDIGO:

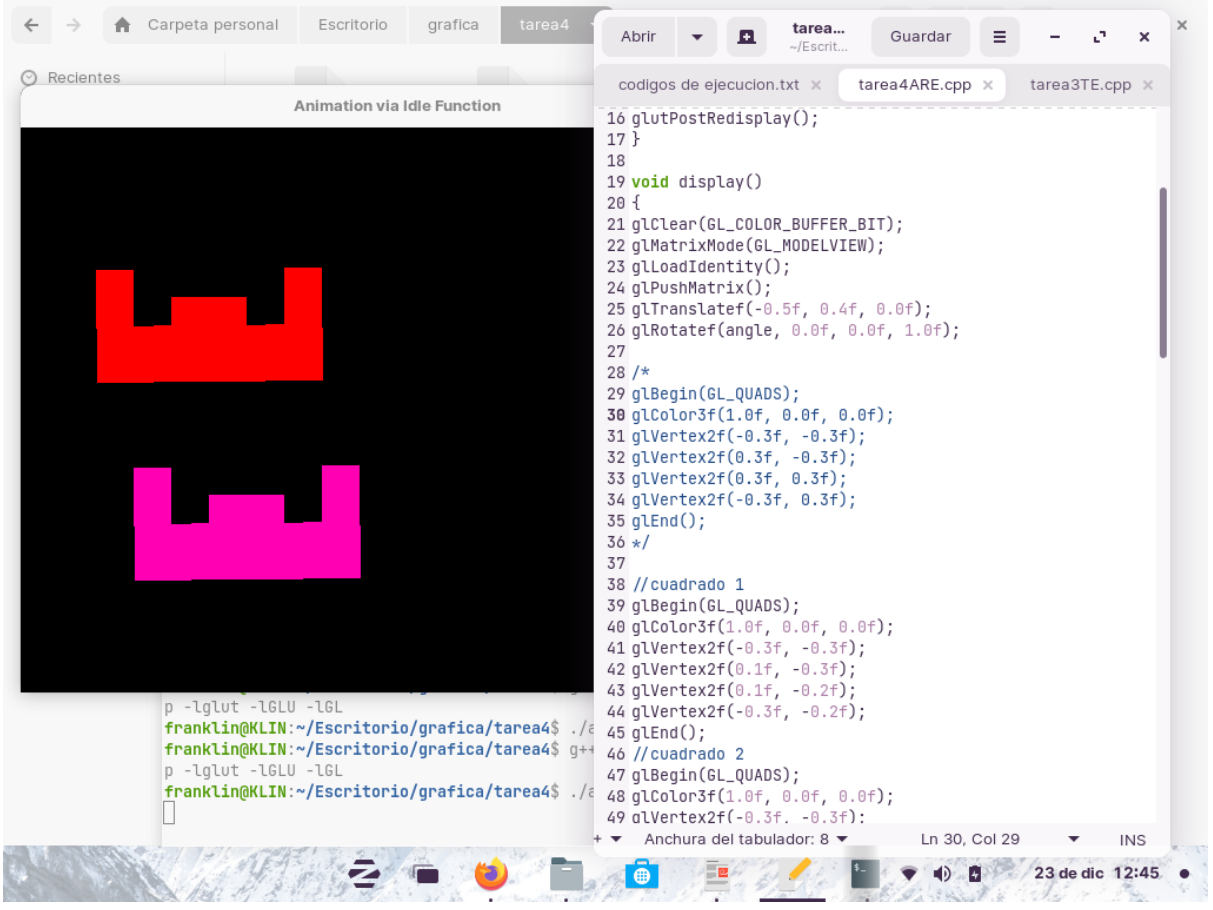
```
#include<GL/glew.h>
#include<math.h>
#include<cstdlib>
#include<GL/glew.h>
#include<GL/glut.h>
#include<stdio.h>
GLfloat angle = 0.0f;


void initGL()
```

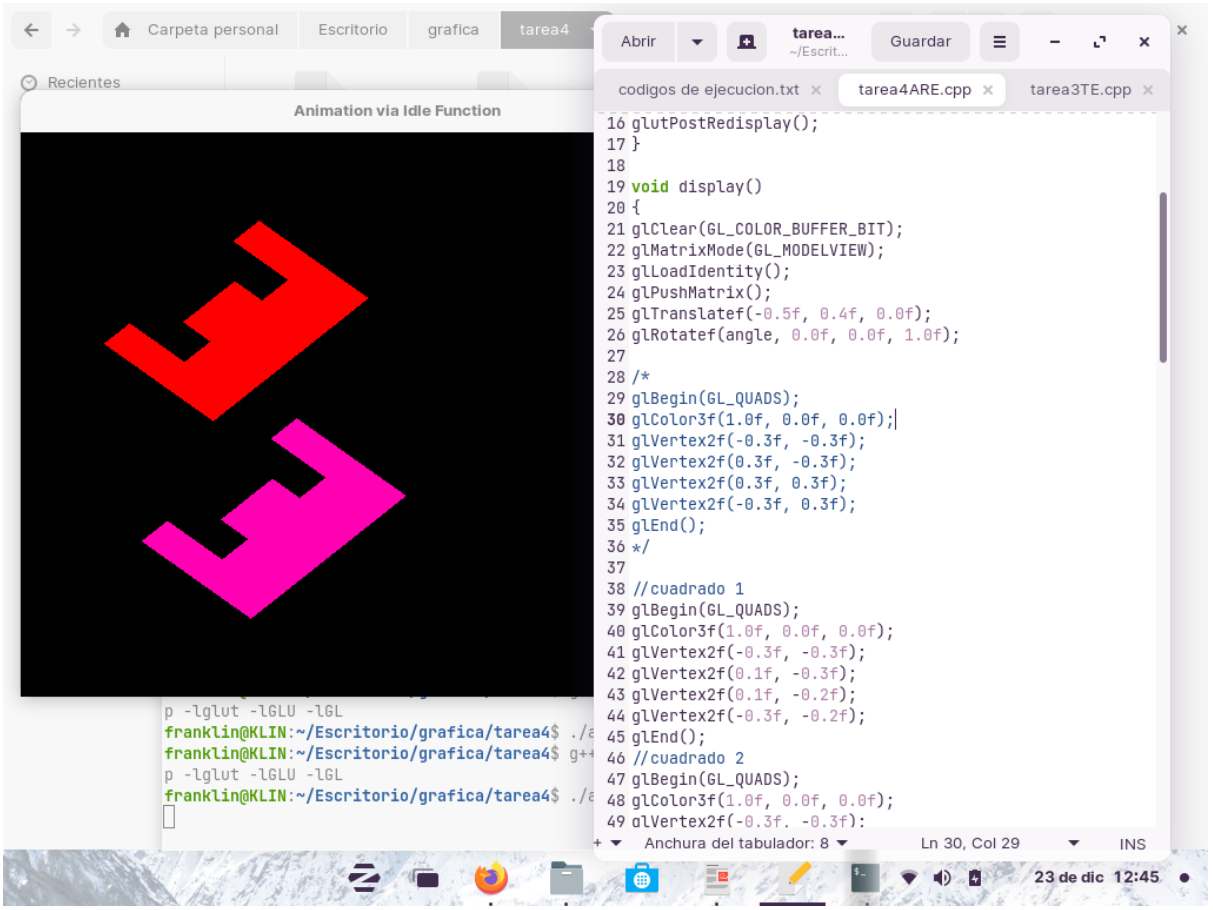
```
{
glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
}
void idle()
{
glutPostRedisplay();
}

void display()
{
glClear(GL_COLOR_BUFFER_BIT);
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glPushMatrix();
glTranslatef(-0.5f, 0.4f, 0.0f);
glRotatef(angle, 0.0f, 0.0f, 1.0f);

/*
glBegin(GL_QUADS);
glColor3f(1.0f, 0.0f, 0.0f);
glVertex2f(-0.3f, -0.3f);
glVertex2f(0.3f, -0.3f);
glVertex2f(0.3f, 0.3f);
glVertex2f(-0.3f, 0.3f);
glEnd();
*/

//cuadrado 1
glBegin(GL_QUADS);
glColor3f(1.0f, 0.0f, 0.0f);
glVertex2f(-0.3f, -0.3f);
glVertex2f(0.1f, -0.3f);
glVertex2f(0.1f, -0.2f);
glVertex2f(-0.3f, -0.2f);
glEnd();
//cuadrado 2
glBegin(GL_QUADS);
glColor3f(1.0f, 0.0f, 0.0f);
glVertex2f(-0.3f, -0.3f);
glVertex2f(-0.1f, -0.3f);
glVertex2f(-0.1f, 0.3f);
glVertex2f(-0.3f, 0.3f);
glEnd();
//cuadrado 3
```

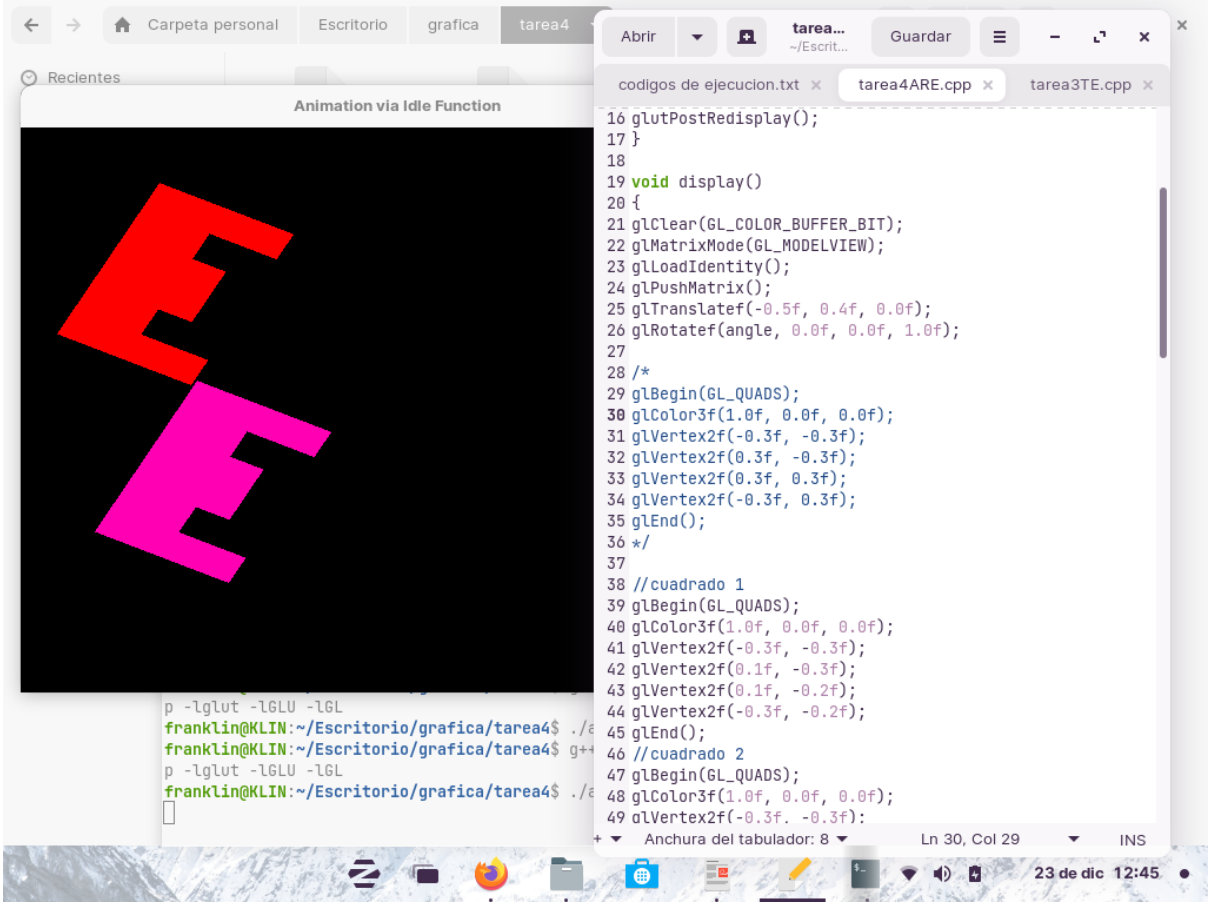
```
glBegin(GL_QUADS);
glColor3f(1.0f, 0.0f, 0.0f);
glVertex2f(-0.3f, 0.3f);
glVertex2f(0.1f, 0.3f);
glVertex2f(0.1f, 0.2f);
glVertex2f(-0.3f, 0.2f);
glEnd();
//cuadrado 4
glBegin(GL_QUADS);
glColor3f(1.0f, 0.0f, 0.0f);
glVertex2f(-0.1f, 0.1f);
glVertex2f(0.0f, 0.1f);
glVertex2f(0.0f, -0.1f);
glVertex2f(-0.1f, -0.1f);
glEnd();


glPopMatrix();
glPushMatrix();
glTranslatef(-0.4f, -0.3f, 0.0f);
glRotatef(angle, 0.0f, 0.0f, 1.0f);
/*
glBegin(GL_QUADS);
glColor3f(0.0f, 1.0f, 0.0f);
glVertex2f(-0.3f, -0.3f);
glVertex2f(0.3f, -0.3f);
glVertex2f(0.3f, 0.3f);
glVertex2f(-0.3f, 0.3f);
glEnd();
*/
//cuadrado 1
glBegin(GL_QUADS);
glColor3f(1.0f, 0.0f, 0.7f);
glVertex2f(-0.3f, -0.3f);
glVertex2f(0.1f, -0.3f);
glVertex2f(0.1f, -0.2f);
glVertex2f(-0.3f, -0.2f);
glEnd();
//cuadrado 2
glBegin(GL_QUADS);
glColor3f(1.0f, 0.0f, 0.7f);
glVertex2f(-0.3f, -0.3f);
glVertex2f(-0.1f, -0.3f);
glVertex2f(-0.1f, 0.3f);
```

```
glVertex2f(-0.3f, 0.3f);
glEnd();
//cuadrado 3
glBegin(GL_QUADS);
glColor3f(1.0f, 0.0f, 0.7f);
glVertex2f(-0.3f, 0.3f);
glVertex2f(0.1f, 0.3f);
glVertex2f(0.1f, 0.2f);
glVertex2f(-0.3f, 0.2f);
glEnd();
//cuadrado 4
glBegin(GL_QUADS);
glColor3f(1.0f, 0.0f, 0.7f);
glVertex2f(-0.1f, 0.1f);
glVertex2f(0.0f, 0.1f);
glVertex2f(0.0f, -0.1f);
glVertex2f(-0.1f, -0.1f);
glEnd();

glPopMatrix();
glutSwapBuffers();
angle += 0.2f;
}

int main(int argc, char** argv)
{
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_DOUBLE);
glutInitWindowSize(640, 480);
glutInitWindowPosition(50, 50);
glutCreateWindow("Animation via Idle Function");
glutDisplayFunc(display);
glutIdleFunc(idle);
initGL();
glutMainLoop();
return 0;
}
```


Carpeta personal
Escritorio
grafica
tarea4
Recientes
Animation via Idle Function
Abrir
tarea...
~/Escrit...
Guardar
codigos de ejecucion.txt
tarea4ARE.cpp
tarea3TE.cpp
16 glutPostRedisplay();
17 }
18
19 void display()
20 {
21 glClear(GL_COLOR_BUFFER_BIT);
22 glMatrixMode(GL_MODELVIEW);
23 glLoadIdentity();
24 glPushMatrix();
25 glTranslatef(-0.5f, 0.4f, 0.0f);
26 glRotatef(angle, 0.0f, 0.0f, 1.0f);
27
28 /*
29 glBegin(GL_QUADS);
30 glColor3f(1.0f, 0.0f, 0.0f);
31 glVertex2f(-0.3f, -0.3f);
32 glVertex2f(0.3f, -0.3f);
33 glVertex2f(0.3f, 0.3f);
34 glVertex2f(-0.3f, 0.3f);
35 glEnd();
36 */
37
38 //cuadrado 1
39 glBegin(GL_QUADS);
40 glColor3f(1.0f, 0.0f, 0.0f);
41 glVertex2f(-0.3f, -0.3f);
42 glVertex2f(0.1f, -0.3f);
43 glVertex2f(0.1f, -0.2f);
44 glVertex2f(-0.3f, -0.2f);
45 glEnd();
46 //cuadrado 2
47 glBegin(GL_QUADS);
48 glColor3f(1.0f, 0.0f, 0.0f);
49 glVertex2f(-0.3f, -0.3f);
p -lglut -lGLU -lGL
franklin@KLIN:~/Escritorio/grafica/tarea4$
franklin@KLIN:~/Escritorio/grafica/tarea4$
p -lglut -lGLU -lGL
franklin@KLIN:~/Escritorio/grafica/tarea4$
Anchura del tabulador: 8
Ln 30, Col 29
INS
23 de dic 12:45


Carpeta personal
Escritorio
grafica
tarea4
Recientes
Animation via Idle Function
Abrir
tarea...
~/Escrit...
Guardar
codigos de ejecucion.txt
tarea4ARE.cpp
tarea3TE.cpp
16 glutPostRedisplay();
17 }
18
19 void display()
20 {
21 glClear(GL_COLOR_BUFFER_BIT);
22 glMatrixMode(GL_MODELVIEW);
23 glLoadIdentity();
24 glPushMatrix();
25 glTranslatef(-0.5f, 0.4f, 0.0f);
26 glRotatef(angle, 0.0f, 0.0f, 1.0f);
27
28 /*
29 glBegin(GL_QUADS);
30 glColor3f(1.0f, 0.0f, 0.0f);
31 glVertex2f(-0.3f, -0.3f);
32 glVertex2f(0.3f, -0.3f);
33 glVertex2f(0.3f, 0.3f);
34 glVertex2f(-0.3f, 0.3f);
35 glEnd();
36 */
37
38 //cuadrado 1
39 glBegin(GL_QUADS);
40 glColor3f(1.0f, 0.0f, 0.0f);
41 glVertex2f(-0.3f, -0.3f);
42 glVertex2f(0.1f, -0.3f);
43 glVertex2f(0.1f, -0.2f);
44 glVertex2f(-0.3f, -0.2f);
45 glEnd();
46 //cuadrado 2
47 glBegin(GL_QUADS);
48 glColor3f(1.0f, 0.0f, 0.0f);
49 glVertex2f(-0.3f, -0.3f);
p -lglut -lGLU -lGL
franklin@KLIN:~/Escritorio/grafica/tarea4$
franklin@KLIN:~/Escritorio/grafica/tarea4$
p -lglut -lGLU -lGL
franklin@KLIN:~/Escritorio/grafica/tarea4$
Anchura del tabulador: 8
Ln 30, Col 29
INS
23 de dic 12:45


Carpeta personal
Escritorio
grafica
tarea4
Recientes
Animation via Idle Function
Abrir
tarea...
~/Escrit...
Guardar
codigos de ejecucion.txt
tarea4ARE.cpp
tarea3TE.cpp
16 glutPostRedisplay();
17 }
18
19 void display()
20 {
21 glClear(GL_COLOR_BUFFER_BIT);
22 glMatrixMode(GL_MODELVIEW);
23 glLoadIdentity();
24 glPushMatrix();
25 glTranslatef(-0.5f, 0.4f, 0.0f);
26 glRotatef(angle, 0.0f, 0.0f, 1.0f);
27
28 /*
29 glBegin(GL_QUADS);
30 glColor3f(1.0f, 0.0f, 0.0f);
31 glVertex2f(-0.3f, -0.3f);
32 glVertex2f(0.3f, -0.3f);
33 glVertex2f(0.3f, 0.3f);
34 glVertex2f(-0.3f, 0.3f);
35 glEnd();
36 */
37
38 //cuadrado 1
39 glBegin(GL_QUADS);
40 glColor3f(1.0f, 0.0f, 0.0f);
41 glVertex2f(-0.3f, -0.3f);
42 glVertex2f(0.1f, -0.3f);
43 glVertex2f(0.1f, -0.2f);
44 glVertex2f(-0.3f, -0.2f);
45 glEnd();
46 //cuadrado 2
47 glBegin(GL_QUADS);
48 glColor3f(1.0f, 0.0f, 0.0f);
49 glVertex2f(-0.3f, -0.3f);
p -lglut -lGLU -lGL
franklin@KLIN:~/Escritorio/grafica/tarea4$
franklin@KLIN:~/Escritorio/grafica/tarea4$
p -lglut -lGLU -lGL
franklin@KLIN:~/Escritorio/grafica/tarea4$
Anchura del tabulador: 8
Ln 30, Col 29
INS
23 de dic 12:45


Carpeta personal
Escritorio
grafica
tarea4
Recientes
Animation via Idle Function
Abrir
tarea...
~/Escrit...
Guardar
codigos de ejecucion.txt
tarea4ARE.cpp
tarea3TE.cpp
16 glutPostRedisplay();
17 }
18
19 void display()
20 {
21 glClear(GL_COLOR_BUFFER_BIT);
22 glMatrixMode(GL_MODELVIEW);
23 glLoadIdentity();
24 glPushMatrix();
25 glTranslatef(-0.5f, 0.4f, 0.0f);
26 glRotatef(angle, 0.0f, 0.0f, 1.0f);
27
28 /*
29 glBegin(GL_QUADS);
30 glColor3f(1.0f, 0.0f, 0.0f);
31 glVertex2f(-0.3f, -0.3f);
32 glVertex2f(0.3f, -0.3f);
33 glVertex2f(0.3f, 0.3f);
34 glVertex2f(-0.3f, 0.3f);
35 glEnd();
36 */
37
38 //cuadrado 1
39 glBegin(GL_QUADS);
40 glColor3f(1.0f, 0.0f, 0.0f);
41 glVertex2f(-0.3f, -0.3f);
42 glVertex2f(0.1f, -0.3f);
43 glVertex2f(0.1f, -0.2f);
44 glVertex2f(-0.3f, -0.2f);
45 glEnd();
46 //cuadrado 2
47 glBegin(GL_QUADS);
48 glColor3f(1.0f, 0.0f, 0.0f);
49 glVertex2f(-0.3f, -0.3f);
p -lglut -lGLU -lGL
franklin@KLIN:~/Escritorio/grafica/tarea4$
franklin@KLIN:~/Escritorio/grafica/tarea4$
p -lglut -lGLU -lGL
franklin@KLIN:~/Escritorio/grafica/tarea4$
Anchura del tabulador: 8
Ln 30, Col 29
INS
23 de dic 12:45