```
//Semestre 2020 - 1
//************************************************************//
//************************************************************//
//************** Alumno (s): *********************************//
//*************            Padilla Herrera Carlos Ignacio                    ******//
//*************                                                                                                                  ******//
//************************************************************//
#include "Main.h"
void InitGL ( GLvoid )     // Inicializamos parametros
{
	glClearColor(0.0f, 0.0f, 0.0f, 0.0f);                                        // Negro de fondo

}

void display(void)   // Creamos la funcion donde se dibuja
{
	glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);        // Limpiamos pantalla y Depth Buffer
	glMatrixMode(GL_MODELVIEW);
	glLoadIdentity();                                                                                                    // Reinicializamos la
actual matriz Modelview

	//Poner aqui codigo ha dibujar
	glPointSize(3.0f);



	//Primera Letra C

		glBegin(GL_POLYGON);
		glColor3f(0.00f, 1.00f, 1.00f);
		glVertex3f(0.50f, 0.50f, -1.0f);
		glVertex3f(1.0f, 0.5f, -1.0f);
		glVertex3f(1.0f, 3.5f, -1.0f);
		glVertex3f(0.5f, 3.5f, -1.0f);
		glEnd();

		glBegin(GL_POLYGON);
		glColor3f(1.00f, 0.20f, 1.00f);
		glVertex3f(1.0f, 0.5f, -1.0f);
		glVertex3f(2.0f, 0.5f, -1.0f);
		glVertex3f(2.0f, 1.0f, -1.0f);
		glVertex3f(1.0f, 1.0f, -1.0f);
		glEnd();

		glBegin(GL_POLYGON);
		glColor3f(1.00f, 0.30f, 1.00f);
		glVertex3f(1.0f, 3.0f, -1.0f);
		glVertex3f(2.0f, 3.0f, -1.0f);
		glVertex3f(2.0f, 3.5f, -1.0f);
		glVertex3f(1.0f, 3.5f, -1.0f);
		glEnd();

		//Segunda Letra C

		glBegin(GL_POLYGON);
		glColor3f(0.00f, 0.50f, 1.00f);
		glVertex3f(2.50f, 0.50f, -1.00f);
```

```
glVertex3f(3.00f, 0.50f, -1.00f);
glVertex3f(3.00f, 3.50f, -1.00f);
glVertex3f(2.50f, 3.50f, -1.00f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(0.50f, 1.00f, 1.00f);
glVertex3f(3.00f, 0.50f, -1.00f);
glVertex3f(4.00f, 0.50f, -1.00f);
glVertex3f(4.00f, 1.00f, -1.00f);
glVertex3f(3.00f, 1.00f, -1.00f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(1.00f, 0.00f, 1.00f);
glVertex3f(3.00f, 3.00f, -1.00f);
glVertex3f(4.00f, 3.00f, -1.00f);
glVertex3f(4.00f, 3.50f, -1.00f);
glVertex3f(3.00f, 3.50f, -1.00f);
glEnd();


//Tercera Letra P

glBegin(GL_POLYGON);
glColor3f(1.00f, 0.20f, 1.00f);
glVertex3f(4.50f, 0.50f, -1.00f);
glVertex3f(5.00f, 0.50f, -1.00f);
glVertex3f(5.00f, 3.50f, -1.00f);
glVertex3f(4.50f, 3.50f, -1.00f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(1.00f, 0.00f, 0.50f);
glVertex3f(5.00f, 1.75f, -1.00f);
glVertex3f(6.25f, 1.75f, -1.00f);
glVertex3f(6.25f, 2.25f, -1.00f);
glVertex3f(5.00f, 2.25f, -1.00f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(0.50f, 0.35f, 1.00f);
glVertex3f(5.75f, 2.25f, -1.00f);
glVertex3f(6.25f, 2.25f, -1.00f);
glVertex3f(6.25f, 3.00f, -1.00f);
glVertex3f(5.75f, 3.00f, -1.00f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(1.00f, 0.99f, 1.00f);
glVertex3f(5.00f, 3.00f, -1.00f);
glVertex3f(6.25f, 3.00f, -1.00f);
glVertex3f(6.25f, 3.50f, -1.00f);
glVertex3f(5.00f, 3.50f, -1.00f);
glEnd();

//Figura estrella
```

```
                glBegin(GL_QUADS);
                glColor3f(1.00f, 1.00f, 0.00f);
                glVertex3f(3.00f,5.00f,-1.00f);
                glVertex3f(3.50f,5.50f,-1.00f);
                glVertex3f(4.00f,5.00f,-1.00f);
                glVertex3f(3.50f,6.50f,-1.00f);
                glEnd();

                glBegin(GL_QUADS);
                glColor3f(1.00f, 1.00f, 0.00f);
                glVertex3f(3.00f, 6.50f, -1.00f);
                glVertex3f(3.50f, 6.00f, -1.00f);
                glVertex3f(4.00f, 6.50f, -1.00f);
                glVertex3f(3.50f, 5.00f, -1.00f);


                glEnd();



        glEnd();



        glFlush();
}

void reshape ( int width , int height )   // Creamos funcion Reshape
{
        if (height==0)                                                                                          //
Prevenir division entre cero
        {
                height=1;
        }

        glViewport(0,0,width,height);

        glMatrixMode(GL_PROJECTION);                                                    // Seleccionamos Projection
Matrix
        glLoadIdentity();

        // Ortogonal
        glOrtho(-10,10,-10,10,0.1,2);

        glMatrixMode(GL_MODELVIEW);                                                             // Seleccionamos
Modelview Matrix
        glLoadIdentity();
}

// Termina la ejecucion del programa cuando se presiona ESC
void keyboard(unsigned char key, int x, int y)
{
         switch (key)
         {
                case 27: exit(0);
                                        break;
```

```
            }
             glutPostRedisplay();
}

int main ( int argc, char** argv )   // Main Function
{
  glutInit            (&argc, argv); // Inicializamos OpenGL
  glutInitDisplayMode (GLUT_RGBA | GLUT_SINGLE); // Display Mode (Clores RGB y alpha | Buffer Sencillo )
  glutInitWindowSize  (300, 300);             // Tamaño de la Ventana
  glutInitWindowPosition (0, 0);//Posicion de la Ventana
  glutCreateWindow    ("Practica 2 2020-1"); // Nombre de la Ventana
  InitGL ();                                                    // Parametros iniciales de la aplicacion
  glutDisplayFunc     ( display );  //Indicamos a Glut función de dibujo
  glutReshapeFunc     ( reshape );          //Indicamos a Glut función en caso de cambio de tamano
  glutKeyboardFunc    ( keyboard );         //Indicamos a Glut función de manejo de teclado
  glutMainLoop        ( );          //

  return 0;
}
```