```
//Semestre 2020 - 1
//************************************************************//
//************************************************************//
//************** Alumno (s): *********************************//
//*************    Padilla Herrera Carlos Ignacio                    ******//
//*************                                                                                    ******//
//************************************************************//
#include "Main.h"

float transZ = -5.0f;
float rotY = 0.0f;
int screenW = 0.0;
int screenH = 0.0;

float red[3] = { 1.0, 0.0, 0.0 };
float green[3] = { 0.0,1.0,0.0 };
float blue[3] = { 0.0,0.0,1.0 };
float white[3] = { 1.0,1.0,1.0 };

void InitGL ( void )     // Inicializamos parametros
{

	//glShadeModel(GL_SMOOTH);                                                    //
Habilitamos Smooth Shading
	glClearColor(0.0f, 0.0f, 0.0f, 0.0f);                          // Negro de fondo
	glClearDepth(1.0f);                                                                //
Configuramos Depth Buffer
	glEnable(GL_DEPTH_TEST);                                                    //
Habilitamos Depth Testing
	glDepthFunc(GL_LEQUAL);                                                    // Tipo
de Depth Testing a realizar
	glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
}

void prisma(float color[3])
{
	GLfloat vertice [8][3] = {
				{0.5 ,-0.5, 0.5},    //Coordenadas Vértice 0 V0
				{-0.5 ,-0.5, 0.5},    //Coordenadas Vértice 1 V1
				{-0.5 ,-0.5, -0.5},    //Coordenadas Vértice 2 V2
				{0.5 ,-0.5, -0.5},    //Coordenadas Vértice 3 V3
				{0.5 ,0.5, 0.5},    //Coordenadas Vértice 4 V4
				{0.5 ,0.5, -0.5},    //Coordenadas Vértice 5 V5
```

```
                        {-0.5 ,0.5, -0.5},    //Coordenadas Vértice 6 V6
                        {-0.5 ,0.5, 0.5},    //Coordenadas Vértice 7 V7
                        };

glColor3fv(color);
        glBegin(GL_QUADS);//Front
                glVertex3fv(vertice[0]);
                glVertex3fv(vertice[4]);
                glVertex3fv(vertice[7]);
                glVertex3fv(vertice[1]);
        glEnd();

        glBegin(GL_QUADS);//Right
                glVertex3fv(vertice[0]);
                glVertex3fv(vertice[3]);
                glVertex3fv(vertice[5]);
                glVertex3fv(vertice[4]);
        glEnd();

        glBegin(GL_QUADS);//Back
                glVertex3fv(vertice[6]);
                glVertex3fv(vertice[5]);
                glVertex3fv(vertice[3]);
                glVertex3fv(vertice[2]);
        glEnd();

        glBegin(GL_QUADS);  //Left
                glVertex3fv(vertice[1]);
                glVertex3fv(vertice[7]);
                glVertex3fv(vertice[6]);
                glVertex3fv(vertice[2]);
        glEnd();

        glBegin(GL_QUADS);  //Bottom
                glVertex3fv(vertice[0]);
                glVertex3fv(vertice[1]);
                glVertex3fv(vertice[2]);
                glVertex3fv(vertice[3]);
        glEnd();

        glBegin(GL_QUADS);  //Top
                glVertex3fv(vertice[4]);
                glVertex3fv(vertice[5]);
```

```
                glVertex3fv(vertice[6]);
                glVertex3fv(vertice[7]);
        glEnd();
}

void display ( void )   // Creamos la funcion donde se dibuja
{
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);  // Limpiamos pantalla
y Depth Buffer
        glLoadIdentity();

                glTranslatef(0.0f, 0.0f, transZ);
                glRotatef(rotY, 0.0f, 1.0f, 0.0f);

                //Poner Código Aquí.

                glPushMatrix();

                        glScalef(4.0f, 4.5f, 1.0f);
                        prisma(white); // Pecho

                glPopMatrix();


                glPushMatrix();

                        glTranslatef(0.0f, 2.5f, 0.0f); //Nos trasladamos al centro del cuello
                        glScalef(0.5f, 0.5f, 1.0f);
                        prisma(green); // Cuello

                glPopMatrix();


                glPushMatrix();

                        glTranslatef(0.0f, 4.125f, 0.0f);
                        glScalef(2.00f, 2.75f, 1.0f);
                        prisma(red); //Cabeza

                glPopMatrix();

                glPushMatrix();
```

```
        glTranslatef(0.0f,-2.75f,0.0f);
        glScalef(4.0f,1.0f,1.0f);
        prisma(green); //Cadera

glPopMatrix();

glPushMatrix();

        glTranslatef(3.0f, 1.75f, 0.0f);
        glScalef(2.0f, 1.0f, 1.0f);
        prisma(blue); //Brazo derecho

glPopMatrix();

glPushMatrix();
        glTranslatef(3.5f, -0.5f, 0.0f);
        glScalef(1.0f, 3.5f, 1.0f);
        prisma(white); //Brazo derecho
glPopMatrix();

glPushMatrix();

        glTranslatef(-3.0f, 1.75f, 0.0f);
        glScalef(2.0f, 1.0f, 1.0f);
        prisma(blue); //Brazo derecho

glPopMatrix();

glPushMatrix();
        glTranslatef(-3.5f, -0.5f, 0.0f);
        glScalef(1.0f, 3.5f, 1.0f);
        prisma(white); //Brazo derecho
glPopMatrix();


glPushMatrix();
        glTranslatef(-1.375f, -5.25f, 0.0f);
        glScalef(1.5f, 4.0f, 1.0f);
        prisma(white); //Pierna izquierda
glPopMatrix();

glPushMatrix();
        glTranslatef(-1.75f, -7.75f, 0.0f);
```

```
            glScalef(2.5f, 1.0f, 1.0f);
            prisma(blue); //Pie izquierdo
        glPopMatrix();

        glPushMatrix();
        glTranslatef(1.375f, -5.25f, 0.0f);
        glScalef(1.5f, 4.0f, 1.0f);
        prisma(white); //Pierna derecha
        glPopMatrix();

        glPushMatrix();
        glTranslatef(1.75f, -7.75f, 0.0f);
        glScalef(2.5f, 1.0f, 1.0f);
        prisma(blue); //Pie derecho
        glPopMatrix();









        /**
        prisma(red);//Primero


        glPushMatrix();
            glTranslatef(3.5f, 0.0f, 0.0f);
            glScalef(3.0f, 0.5f, 1.0f);//Poner Código Aquí.
            prisma(white);//Segundo
        glPopMatrix(); //detengo el efecto de la escala. Haría que ya no tengo que hacer
las divisiones. Sin tener que hacer operaciones adicionales
        // Ya no me tengo que preocupar por la escala


        glPushMatrix();
            glTranslatef(0.0f, 4.25f, 0.0f);
            glScalef(0.5/3.0f, 2.25/0.5f, 1.0);
            prisma(green); //Tercero
        glPopMatrix();
        **/
```

```
  glutSwapBuffers ( );
  // Swap The Buffers
}

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();

        // Tipo de Vista
        glFrustum (-0.1, 0.1,-0.1, 0.1, 0.1, 50.0);

        glMatrixMode(GL_MODELVIEW);                                     //
Seleccionamos Modelview Matrix
}

void keyboard ( unsigned char key, int x, int y )  // Create Keyboard Function
{
        switch ( key ) {
                case 'w':
                case 'W':
                        transZ +=0.3f;
                        break;
                case 's':
                case 'S':
                        transZ -= 0.3f;
                        break;
                case 'a':
                case 'A':
                        transY += 0.3f;

                        break;
                case 'd':
```

```
                case 'D':
                        transY -= 0.3f;

                        break;

                case 27:        // Cuando Esc es presionado...
                        exit ( 0 );   // Salimos del programa
                break;
                default:        // Cualquier otra
                break;
  }
        glutPostRedisplay();
}

void arrow_keys ( int a_keys, int x, int y )  // Funcion para manejo de teclas especiales (arrow
keys)
{
  switch ( a_keys ) {
    case GLUT_KEY_UP:               // Presionamos tecla ARRIBA...
                break;
    case GLUT_KEY_DOWN:             // Presionamos tecla ABAJO...
                break;
        case GLUT_KEY_LEFT:
                break;
        case GLUT_KEY_RIGHT:
                break;
    default:
      break;
  }
  glutPostRedisplay();
}


int main ( int argc, char** argv )   // Main Function 2020
{
  glutInit            (&argc, argv); // Inicializamos OpenGL
  glutInitDisplayMode (GLUT_RGB | GLUT_DOUBLE | GLUT_DEPTH); // Display Mode (Clores
RGB y alpha | Buffer Doble )
  screenW = glutGet(GLUT_SCREEN_WIDTH);
  screenH = glutGet(GLUT_SCREEN_HEIGHT);
  glutInitWindowSize  (screenW, screenH);   // Tamaño de la Ventana
  glutInitWindowPosition (0, 0);          //Posicion de la Ventana
  glutCreateWindow    ("Practica 4"); // Nombre de la Ventana
```

```
  printf("Resolution H: %i \n", screenW);
  printf("Resolution V: %i \n", screenH);
  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
  glutSpecialFunc     ( arrow_keys );  //Otras
  glutMainLoop        ( );          //

  return 0;
}
```