CÓMO SE EJECUTA

qmake-qt5
make
./examen

1. INICIO

En paintGL() al comienzo de todo pon:
projectTransform();
viewTransform();

2. TG

Tener en cuenta

```
 1 Creacio del VAO       glGenVertexArrays(1, &VAO);
 2 Activacio del VAO     glBindVertexArray(VAO);
 3 Creacio d'un VBO      glGenBuffers(1, &VBO);
 4 Activacio del VBO     glBindBuffer(GL_ARRAY_BUFFER, VBO);
 5 Omplir VBO amb dades      glBufferData(GL_ARRAY_BUFFER,
                    sizeof(Vertices), Vertices, GL_STATIC_DRAW);
 6 Atribut (id) geom (VBO)      glVertexAttribPointer(vertexLoc, 3,
                    GL_FLOAT, GL_FALSE, 0, 0);
 7 Activa atribut       glEnableVertexAttribArray(vertexLoc);
 8 Desactiva el VAO      glBindVertexArray(0);
```

TGS MODELOS

```
void MyGLWidget::modelTransformModel ()
{
 glm::mat4 TG(1.f);
 TG = glm::translate(TG, glm::vec3(8.0,0,0));
 TG = glm::rotate(TG, float(M_PI/2), glm::vec3(0, 1, 0));
 TG = glm::scale(TG, glm::vec3(4*escalaModel, 4*escalaModel, 4*escalaModel));
 TG = glm::translate(TG, -centreBaseModel);

 glUniformMatrix4fv (transLoc, 1, GL_FALSE, &TG[0][0]);
}
```

Si queremos más adelante un **KEYPRESS EVENT:**

1. declaro vector posmodel en .h
2. Lo inicializo en iniEscena() con la posición que el enunciado dice que tiene el modelo.
3. En ese modelo, cuando haga su modelTransform() en la siguiente linea lo sustituyo por posmodel:
TG = glm::translate(TG, **glm::vec3(8.0,0,0)**); → TG = glm::translate(TG, **posmodel**);

ROTACIONES DE LOS EJEX X, Y, Z

Para rotar en sentido del reloj es negativo, else es positivo.

float((M_PI)/1) →180º

float((M_PI)/2) →90º

float((M_PI)/3) →60º

float((M_PI)/4) →45º

DIBUJAR DOS MODELOS CON MISMO VAO

En paintGL() tendras:

glBindVertexArray (VAO_Pil); → se activa

modelTransformPilota (); → se hacen las TG

glDrawArrays(GL_TRIANGLES, 0, pil.faces().size()*3); → se pinta

Para hacer otro modelo añadirias antes del glBindVertexArray(0);

modelTransformPilota2 (); → se hacen las TG

glDrawArrays(GL_TRIANGLES, 0, pil.faces().size()*3); → se pinta

DIBUJAR DOS MODELOS CON DIFERENTE VAO

Declaras nuevo VAO2 en .h

Lo activas en paintGL()

En tu createBuffersVAO2 pones:

glGenVertexArrays(1, &VAO2);

glBindVertexArray(VAO2);

3. CAMARA

CAMARA EN 3A PERSONA – PERSPECTIVA

*** En initialize()**

En initializeGL() tiene que llamar a (si se hace reset solo?)

iniEscena()

iniCamera()

inillum()

*** En iniEscena()**

Tenemos que hacer dos vectores
glm::vec3 pmin = glm::vec3(, ,);
glm::vec3 pmax = glm::vec3(, ,);
posmodel = glm::vec3(, ,); → posicion donde esta el modelo

En el .h declaramos centreEsc, radiEsc
centreEsc = (pmin+pmax)/2.0f;
radiEsc = glm::distance(pmin,pmax)/2.0f;

Para hacer el pmin y el pmax tenemos que ver como esta dibujado el suelo, pared, porteria etc y coger los max y min puntos existentes **MUCHO OJO.**

Cualquier otro objeto también se puede inicializar aquí.

*** En iniCamera()**

En el .h declaramos:
float angleY, angleX, zn, zf,ra, fov, fovini;

En iniCamera ponemos:
angleY = lo que diga el enunciado;
angleX = lo que diga el enunciado;
fov = 2.0*asin(radiEsc/(2*radiEsc));
fovini = fov;
vrp = ((minEsc + maxEsc)/ 2.f); → **aka centreEsc**
ra = 1.0;
zn = radiEsc;
zf = 3*radiEsc;
projectTransform();
viewTransform();

*** En resizeGL()**

Para hacer que no se corte ni deforme:

```
void MyGLWidget::resizeGL (int w, int h)
{
        ra = float(w) / float(h);
        if (ra < 1.0) {
                fov = 2.0*atan(tan(fovini/2.0)/ra);
        }
        else fov = fovini;
}
```

En el resize (si tiene una Óptica **ortogonal y perspectiva**), hacemos:
Declaramos xl, xr, yb, yt en el .h

```
void MyGLWidget::resizeGL (int w, int h)
{
        ra = float(w) / float(h);
        if (ra < 1.0) {
                fov = 2.0*atan(tan(fovini/2.0)/ra);
                xl = -radiEsc;
                xr = radiEsc;
                yb = -radiEsc/ra;
                yt = radiEsc/ra;
        }
        else{
                 fov = fovini;
                xl = -radiEsc;*ra
                xr = radiEsc*ra;
                yb = -radiEsc;
                yt = radiEsc;
        }
}
```

*** En viewTransform()**

Aqui van los angulos de Euler:

```
glm::mat4 View(1.f);
        View= glm::translate(View, glm::vec3(0,0,-2*radiEsc));
        View = glm::rotate(View, angleX, glm::vec3(1, 0, 0));
        View = glm::rotate(View, -angleY, glm::vec3(0, 1, 0));
        View= glm::translate(View, -vrp); → vrp o centreEsc

        glUniformMatrix4fv (viewLoc, 1, GL_FALSE, &View[0][0]);
```

*** En projectTransform()**

```
glm::mat4 Proj = glm::perspective(fov, ra, zn, zf);
glUniformMatrix4fv (projLoc, 1, GL_FALSE, &Proj[0][0]);
```

CAMARA 1ERA PERSONA – ORTOGONAL

Si nos dicen añadir una camara en primera persona que se active o desactive con un elemento de interficie, teniendo ya una camara en 3a persona.

Declaramos un bool primeraperona en el .h
En iniCamera() inicializo primerapersona = false;
En projectTransform() hacemos:

```
void MyGLWidget::projectTransform ()
{
  glm::mat4 Proj;
  if(!primerapersona) Proj = glm::perspective(fov, ra, zn, zf);
    else {
        float fov2 = M_PI/2; → lo dice el enunciado
        float zn2 = 0.01; → suficiente peque para que se vea znear
        float zf2 = 100; → suficiente grande para que se vea toda la escena
        Proj = glm::perspective(fov2,ra,zn2,zf2);
  }
  glUniformMatrix4fv (projLoc, 1, GL_FALSE, &Proj[0][0]);
}
```

Luego en el viewTransform()

```
void MyGLWidget::viewTransform ()
{
        if (! primerapersona) {
          glm::mat4 View(1.f);
          View= glm::translate(View, glm::vec3(0,0,-2*radiEsc));
          View = glm::rotate(View, angleX, glm::vec3(1, 0, 0));
          View = glm::rotate(View, -angleY, glm::vec3(0, 1, 0));
          View= glm::translate(View, -vrp); → vrp puede ser centreEsc.

          glUniformMatrix4fv (viewLoc, 1, GL_FALSE, &View[0][0]);
        }
        else {
          glm::mat4 View(1.f);
          View= glm::lookAt(glm::vec3(8,5,0), glm::vec3(-7,5,0), glm::vec3(0,1,0));
          glUniformMatrix4fv (viewLoc, 1, GL_FALSE, &View[0][0]);
        }
}
```

Si tambien hay camara en 1era persona, delcaramos en .h xl,xr,yb,yt y en iniCamera()
ponemos:(creo que es float)

```
float xl = -radiEsc
float xr = radiEsc
float yb = -radiEsc
float yt = radiEsc
```

PARA EL QT

En .cpp hacemos una funcion:

```
void MyGLWidget::canvicam() {
        makeCurrent();
        primerapersona = !primerapersona;
        viewTransform();
        update();
}
```

4. ILUMINACION

Tenemos dos ficheros, vertexShader y fragmentShader.

*** CALCULO ILU VS PHONG**

Piden CON UNA LUZ BLANCA → colFocus

```
void MyGLWidget::inillum () {
        colFocus = glm::vec3(0.8,0.8,0.8);
    glUniform3fv (colFocusLoc, 1, &colFocus[0]);
}
```

Si dicen: Afegeix a l'escena el càlcul d'il·luminació al Vertex Shader usant el model d'il·luminacio de Phong i amb un focus de llum blanca situat sempre exactament a la posició de la càmera. Mira → escritorio/total/1.porteria/solucion/examen/shaders

Si en:

5. QT

Cambio de camara:

```
void MyGLWidget::segonacamera () {
        makeCurrent();
        segcam = !segcam;
        update();
}
```

6. KEYPRESS

Afegeix la possibilitat que prement la tecla 'F' el focus de llum passi a ser un focus d'escena a la posició (0,10,0) de l'escena. Tornant a prémer tecla 'F' el focus serà de nou el focus de càmera descrit a l'exercici anterior.

Se hace algo asi,

```
case Qt::Key_F: {
      focEsc = !focEsc;
      posicionafocus();
```

```
      break;
    }
```

Cambio de camara con tecla:

```
case Qt::Key_S: {
      segcam = !segcam;
      break;
    }
```

(más cosas en Escritorio/total/2.pelota/examen/examen1819)

En carregaShaders() → focEscLoc = glGetAttribLocation (program->programId(), "focEsc");

Si dicen algo como: Afegeix la possibilitat que el legoman porter, Lego2, es mogui en la direcció de l'eix Z, entre els punts pmin=(-7,0,-3) i pmax=(-7,0,3).
Vamos a keypressevent y hacemos:

```
case Qt::Key_Left: { // moviment de la pilota
      if (posPort.z >= -3) {
        posPort.z += 0.15;
        modelTransformModel2();
            }
      break;
    }
```

7. MOUSE EVENT

En general es asi lo que hay que poner:

```
void MyGLWidget::mouseMoveEvent(QMouseEvent *e)
{
  makeCurrent();
  if (DoingInteractive == ROTATE)
  {
    angleY += (e->x() - xClick) * M_PI / 180.0;
    angleX -= (e->y() - yClick) * M_PI / 180.0;
    viewTransform ();
  }

  xClick = e->x();
  yClick = e->y();
```

```
  update ();
}
```

8. CARGAR MODELO

Si hay createBuffersModel() →
cambiamos el modelo que nos piden en: patr.load("./models/legoman.obj");

Un modelo se declara en el .h como **Model patr**
Cada modelo tiene su createBuffers(), createBufferPilota, createBufferCasa etc.

9. EJEMPLO RESET

Cuando se clique boton reset hace to esto

```
void MyGLWidget::reset() {
        makeCurrent();
        inillum();
        iniciPilota();
        iniEscena();
        iniCamera();
        posPort = glm::vec3(-7,0,0);
        emit sendres(primerap);
        viewTransform();
        projectTransform();
        update();
}
```

OTRA COSA MARIPOSA

```
void MyGLWidget::inillum () {
        colFocus = glm::vec3(0.8,0.8,0.8);
    glUniform3fv (colFocusLoc, 1, &colFocus[0]);
}
```

10. ZOOM