3. Una esfera de radi 1 es visualitza en un *viewport* quadrat de 400 per 400, amb una càmera posicionada correctament amb angles d'Euler, i on el mètode per a definir la projecció de la càmera utilitza la següent crida:

```
gluPerspective(60.0, 1.0, 1.0, 10.0);
```

L'usuari ha redimensionat la finestra a 500 d'amplada per 400 d'alçada. Digues què cal canviar de la càmera per tal que es vegi l'esfera correctament (sense retallar-la ni deformar-la).

a. Incrementar l'angle d'obertura vertical (FOV) i la relació d'aspecte del *window*.
b. Augmentar la relació d'aspecte del *window* i la distància al ZNear.
c. Només augmentar la relació d'aspecte del *window*.
d. Només canviar l'angle d'obertura vertical (FOV).

4. Tenim una esfera de radi 1 centrada al punt (0.0, 0.0, 0.0) i una càmera amb una òptica definida com

```
glMatrixMode/(GL_PROJECTION);
glLoadIdentity();
glOrtho(0.0, 1.0, 0.0, 1.0, 0.0, 3.0);  // left,right,bottom,up,znear,zfar
```

Si el mètode de pintat és:

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glColor3f(1.0, 0.0, 0.2);
glTranslatef(0.0, 0.0, -2.0);
glutSolidSphere(1.0, 20., 20.);
```

Què es veurà?

a. La part superior dreta de l'esfera.
b. La part superior de l'esfera.
c. La part inferior esquerra de l'esfera.
d. La part dreta de l'esfera.

5. Quin dels següents codis creus que permetria definir la transformació de la càmera VRP=(0,1,0), OBS=(20,1,20) i up=(0,1,0) amb transformacions geomètriques?

a.

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0, 0, -20*sqrt(2.));
glRotated(45, 0, 0, 1);
glTranslatef(0, -1, 0);
```

b.

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0, 0, 20);
glRotated(-45, 0, 1, 0);
glTranslatef(0, -1, 0);
```

c.

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0, 0, -20);
glRotated(45, 0, 1, 0);
glTranslatef(0, -1, 0);
```

d.

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0, 0, -20*sqrt(2.));
glRotated(-45, 0, 1, 0);
glTranslatef(0, -1, 0);
```

6. Quan s'inicialitza la càmera, en quin ordre cal indicar les transformacions de càmera i el *viewport* a OpenGL?

a. No importa l'ordre en què s'indiquen.
b. Transformació de posició+orientació, transformació de projecció, *viewport*.
c. La transformació de projecció, transformació de posició+orientació, *viewport*.
d. *Viewport*, tansformació de projecció, transformació de posició+orientació.

Nom i Cognoms: _____Grup: ____DNI: _____

3 (1 punt) Tenim una càmera axonomètrica que permet veure tota una escena sense retallar ni deformar, des de qualsevol punt de vista, amb l'òptica definida amb:

```
glMatrixMode (GL_PROJECTION);
glLoadIdentity();
glOrtho(-10, 10, -2, 2, 1, 30);
```

Modifiquem la relació d'aspecte del *viewport* a 1, quina seria la crida a glOrtho correcta per a continuar veient tota l'escena sense deformacions?

a. **glOrtho(-10, 10, -10, 10, 1, 30);**
b. glOrtho(-10, 10, 1, -1, 1, 30);
c. glOrtho(-2, 2, -2, 2, 1, 30);
d. glOrtho(-5, 5, -5, 5, -5, 5);

4 (1 punt) Disposem d'una càmera ortogonal amb els següents paràmetres: OBS=(0.,0.,0.), VRP=(-1.,0.,0.), up=(0.,1.,0.), window de (-5,5) a (5,5), ra=1, zn=5, zf=10.
Indiqueu quin conjunt de paràmetres d'una càmera perspectiva defineix un volum de visió que conté l'anterior (és a dir, garanteix que es veurà, coma mínim, el mateix que amb la càmera axonomètrica):

a. **FOV= 90, ra=1, zn= 5, zf=10**
b. FOV= 60, ra=1, zn=5, zf=10
c. FOV= 60, ra= 2, zn=6, zf=11
d. FOV= 90, ra= 0.5, zn=5, zf=10

5 (1 punt) Tenim una escena que es pinta de la següent forma:

```
glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
glEnable (GL_DEPTH_TEST);
glMatrixMode(GL_PROJECTION);
glLoadIndentity();
gluPerspective(60.0, 1.0, 1.0, 100.);
glMatrixMode(GL_MODELVIEW);
glTranslatef(0.0, 0.0, -20.0);
glRotated(-90.0, 0.0, 1.0, 0.0);

glColor3f(1.0,0.0,0.0);
glBegin(GL_QUADS);
glVertex3f(0.0, -2.0, -1.0);
glVertex3f(0.0, -2.0, 1.0);
glVertex3f(0.0, 2.0, 1.0);
glVertex3f(0.0, 2.0, -1.0);
glVertex3f(0.0, 2.0, -2.0);
glVertex3f(0.0, 2.0, 2.0);
glVertex3f(0.0, 4.0, 2.0);
glVertex3f(0.0, 4.0, -2.0);
glEnd();
```

Digues què es veu:

a. Una espècie de *T* invertida formada per dos rectangles de color vermell amb el seu centre una mica per sobre del centre de la pantalla.
b. Dos rectangles, un vertical i un d'horitzontal que formen una espècie de *L* centrada en la pantalla una mica cap a l'esquerra.
c. Una línia vermella que va des del centre de la pantalla fins una mica més amunt sense arribar al límit superior del *viewport.*
d. **Una espècie de *T* formada per dos rectangles de color vermell amb el seu centre una mica per sobre del centre de la pantalla.**

6 (1 punt) Per a posicionar una càmera perspectiva a una determinada distància del VRP i en una determinada orientació, el codi OpenGL ha de realitzar una sèrie de crides de transformacions geomètriques. Digues quina de les combinacions següents de crides a rotacions i translacions conté les crides necessàries i està en l'ordre correcte:

a. Rotació respecte X, rotació respecte Z, rotació respecte Y, translació –VRP.
b. Rotació respecte Z, rotació respecte Y, rotació respecte X, translació –VRP i translació en Z –distància.
c. **Translació en Z –distància, rotació respecte Z, rotació respecte X, rotació respecte Y, translació –VRP.**
d. Rotació respecte Z, rotació respecte Y, rotació respecte X, translació –VRP.

Nom i Cognoms: _____Grup: ____DNI: _____

3 (1 punt) Disposem d'una càmera axonomètrica amb els següents paràmetres: OBS=(0.,0.,0.), VRP=(-1.,0.,0.), up=(0.,1.,0.), *window* de (-5,-5) a (5,5), zn=5, zf=10.
Indiqueu quin altre conjunt de paràmetres de càmera defineix exactament el mateix volum de visió (és a dir, garanteix generar exactament la mateixa imatge de l'escena):

a. **OBS= (1,0,0), VRP= (0,0,0), up=(0,2,0), zn= 6, zf=11**
b. OBS= (0,1,0), VRP=(0,0,0), up= (0,1,0), zn=5, zf=10
c. OBS= (0,0,0), VRP=(-2,0,0), up=(0,1,0), zn=6, zf=11
d. OBS= (-1,0,0), VRP=(0,0,0), up=(0,1,0), zn=-1, zf=9

4 (1 Punt) Tenim el següent codi que pinta una escena:

```
glViewport (0,0,600,600);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluPerspective(60., 1.0, 1.0, 100.0);
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt(0., 0.,10,0,0,0,0,1,0);
glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
glEnable (GL_DEPTH_TEST);
glColor3f(1,0,0);
glScaled(1.0, 5.0, 1.0);
glRotatef(-90, 1.0, 0.0, 0.0);
glRotatef(-45, 0.0, 1.0, 0.0);
glScaled(1.0, 5.0, 1.0);
glutSolidCube(1.0);
glRotatef(30., 0, 0, 1.);
```

Digues què es veu:

a. Un triangle vermell amb la seva base horitzontal al centre de la pantalla i la punta superior cap amunt.
b. Cap de les altres
c. Un rombe més ample que alt amb el seu centre situat al centre de la pantalla.
d. Un rombe més alt que ample amb el seu centre situat al centre de la pantalla.

5 (1 Punt) Quan es realitza la crida a glVertex3f(x,y,z), OpenGL realitza una sèrie de transformacions per a obtenir el píxel en què cal pintar-lo. Quina d'aquestes seqüències es correspon amb les transformacions que es fan?

a. Transformació de *modelview*, transformació de projecció, transformació *window-viewport*.
b. Transformació de projecció, transformació *window-viewport*, transformació de *modelview*.
c. Transformació de projecció, transformació de *modelview*, transformació de *window-viewport*.
d. Transformació de *window-viewport*, transformació de projecció, transformació de *modelview*.

6 (1 punt) En les inicialitzacions prèvies al pintat d'una escena, tenim la següent seqüència d'instruccions OpenGL que defineix una càmera amb un *window* quadrat i un *viewport* també quadrat:

```
gluPerspective(myFovy, 1.0, myNear, myFar);
glViewport (0, 0, 400, 400);
```

quina diferència s'observaria en la visualització de l'escena si les canviem per:

```
gluPerspective (myFovy, 2.0, myNear, myFar);
glViewport (0, 0, 400, 400);
```

a. Cap perquè la relació d'aspecte de la càmera és superior a la del *viewport*, per això no cal modificar res més.
b. L'escena es veurà deformada amb el doble de llargada que amplada.
c. L'escena es veurà deformada amb el doble d'amplada que llargada.

d. Com no hem modificat FOV, es veurà retallada l'escena respecte l'inicial.

4. *(1 punt)* Tenim una escena com la de la pregunta 1 però amb la base (de la pila de cubs) centrada en el punt (X, 0, Z). Volem una càmera que miri els cubs en una vista en planta (des de dalt) i els vegi centrats a la vista. Suposant l'òptica de la càmera ben definida, indica quin dels següents trossos de codi et permetria definir amb transformacions geomètriques la posició i orientació de la càmera.

a)
```
glMatrixMode (GL_PROJECTION);
glLoadIdentity ();
glTranslatef (0, 0, -5);
glRotatef (-90, 0, 0, 1);
glRotatef (90, 1, 0, 0);
glRotatef (-90, 0, 1, 0);
glTranslatef (X, 0, Z);
```

b)
```
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ():
glTranslatef (0, 0, -5);
glRotatef (90, 1, 0, 0);
glRotatef (-90, 0, 1, 0);
```

c)
```
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ():
glTranslatef (0, 0, -5);
glRotatef (90, 0, 0, 1);
glRotatef (-90, 1, 0, 0);
glRotatef (-90, 0, 1, 0);
glTranslatef (-X, 0, -Z);
```

d)
```
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ():
glTranslatef (0, 0, -3);
glRotatef (90, 0, 0, 1);
glRotatef (90, 1, 0, 0);
glTranslatef (-X, -2, -Z);
```

5. *(1 punt)* Tenim definida una càmera axonomètrica amb paràmetres: OBS = (0, -1, 0), VRP = (0, 0, 0); Vup = (0, 0, 1), Window = (-2, 2, -2, 2), ZNear = 1, ZFar = 4, i una relació d'aspecte de la vista de 1 (rav = 1). Indica quina de les següents càmeres perspectiva seria adient per a definir un volum de visió **que inclogui completament** l'anterior:

a) OBS=(0, -2, 0), VRP=(0, 2, 0), Vup=(0, 0, 1), FOV=90, ra=1, ZNear=2, ZFar=5.
b) OBS=(0, 2, 0), VRP=(0, 0, 0), Vup=(0, 1, 0), FOV=90, ra=1, ZNear=1, ZFar=4.
c) OBS=(0, -1, 0), VRP=(0, 0, 0), Vup=(0, 0, 1), FOV=90, ra=1, ZNear=1, ZFar=4.
d) OBS=(0, -2, 0), VRP=(0, 0, 0), Vup=(0, 0, 1), FOV=60, ra=1, ZNear=1, ZFar=4.

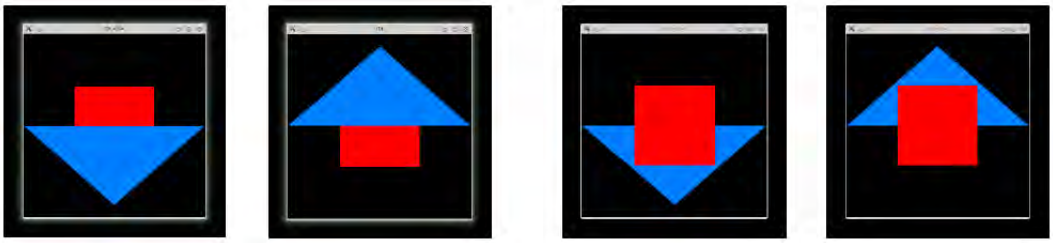
6. *(1 punt)* Tenim definida correctament una càmera perspectiva que ens permet veure una escena completa sense retallar i sense deformació. Si a aquesta càmera li modifiquem el FOV, indica què caldrà modificar també si no volem que hi hagi deformació:

a) La relació d'aspecte de la vista (rav).
b) No cal modificar res més.
c) La relació d'aspecte del window (raw).
d) Depèn de si incrementem o decrementem el FOV.

7. *(1 punt)* Pintem una escena amb el següent codi:

```
glViewport (0, 0, 800, 800);
glEnable (GL_DEPTH_TEST);
glClearColor (0, 0, 0, 1);
glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
glMatrixMode (GL_PROJECTION); // Inici definició de càmera
glLoadIdentity ();
glOrtho (-5, 5, -5, 5, 5, 15);
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ();
glTranslatef (0, 0, -10);
glRotatef (180, 0, 0, 1);
glRotatef (90, 1, 0, 0);  // Fí definició de càmera
glColor3f (0, 0.5, 1);
glutSolidCone (5,5,20,20); // radi, alçada, orientació Z+, base centrada en (0,0,0)
glColor3f (1, 0, 0);
glPushMatrix ();
glTranslatef (0, -2, 0);
glutSolidCube (4); // costat 4, centrat a l'origen
glPopMatrix ();
```

Digues quina de les imatges següents es veu:



a) La primera imatge

b) La segona imatge

c) La tercera imatge

d) La quarta imatge

8. *(1 punt)* Tenim una esfera de radi 3 centrada a l'origen de coordenades i un focus de llum situat a la posició (0, 3, 5) de color (1, 1, 0). No hi ha llum ambient. Un observador es mira aquesta escena des de la posició (0, 0, 5) i mirant cap al centre de l'esfera, i el que observa és una esfera que té una part propera a la silueta per la part de baix de l'esfera de color negre, un degradat de colors verds que són més clars per la part de dalt de l'esfera i més foscos per la part de baix i una taca de color groc cap a la part del mig de la semiesfera superior. Quines constants de material de l'esfera permeten que es pugui veure aquesta escena de la forma descrita?

a) Ka = (0, 0.2, 0), Kd = (0, 0.8, 0), Ks = (0, 0, 0) i N = 100

b) Ka = (0.2, 0.2, 0.2), Kd = (0.8, 0.8, 0.8), Ks = (1, 1, 1) i N = 100

c) Ka = (0, 0.2, 0.2), Kd = (0, 0,8, 0.8), Ks = (1, 1, 1) i N = 100

d) Ka = (0, 0.2, 0.2), Kd = (0, 0.8, 0.8), Ks = (0, 1, 1) i N = 100

3. *(1 punt)* Per a visualitzar l'escena de l'exercici 1, un estudiant proposa la següent càmera: OBS=(20,0,0), VRP=(-9,0,0), up=(0,0,1) , ZN=15, ZF=40, FOV=90º i ra=1. Quan pinta l'escena no veu res en pantalla (malgrat que la rutina `pintaEscena()` és correcta). Quin paràmetre de la càmera creus que no és el correcte?

   Considerem les mides de les capses de la vaca i del Patricio, un cop ubicats, iguals que en l'exercici 2.

   a) vector up
   b) ZN
   c) ra
   d) VRP

   Solució: b)

4. *(1 punt)* Cal definir una càmera a OpenGL; quin dels següents pseudocodis és correcte? Noteu que tant sols canvia l'ordre en què es fan les crides.

   1)
   ```
   VM=lookAT(OBS, VRP, up)
   viewMatrix (VM)
   PM=perspective (FOV, ra, zn,zf)
   projectionMatrix(PM)
   glViewport(...)
   modelMatrix(TG)
   pintaescena()
   ```

   2)
   ```
   modelMatrix(TG)
   PM=perspective (FOV, ra, zn,zf)
   projectionMatrix(PM)
   VM=lookAT(OBS, VRP, up)
   viewMatrix (VM)
   glViewport(...)
   pintaescena()
   ```

   3)
   ```
   VM=lookAT(OBS, VRP, up)
   viewMatrix (VM)
   PM=perspective (FOV, ra, zn,zf)
   projectionMatrix(PM)
   modelMatrix(TG)
   glViewport(...)
   pintaescena()
   ```

   4)
   ```
   glViewport(...)
   VM=lookAT(OBS, VRP, up)
   viewMatrix (VM)
   PM=perspective (FOV, ra, zn,zf)
   projectionMatrix(PM)
   modelMatrix(TG)
   pintaescena()
   ```

   a) només 1) i 4) són correctes
   b) només 4) és correcte
   c) tots són correctes
   d) tots són correctes menys 2)

   Solució: c)

5. *(1 punt)* Imagina que tenim l'escena de la vaca + Patricio de la pregunta 1 i els volem girar entorn l'eix Y (com si es tractés d'una peça d'uns cavallets –*"tiovivo"*–). Suposant que TG1 és la matriu de TG per ubicar la vaca i TG2 és la matriu de TG per ubicar el Patricio quin dels següents codis és correcte?

a)
```
AUX= Rotate(alfa,0,1,0)
TG1= AUX*TG1
TG2= AUX*TG2
modelMatrix(TG1)
pintaVaca()
modelMatrix(TG2)
pintaPatricio()
```

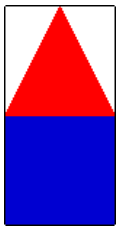
b)
```
modelMatrix(TG1)
pintaVaca()
Rotate (alfa,0,1,0)
modelMatrix(TG2)
pintaPatricio()
Rotate (alfa,0,1,0)
```

c)
```
AUX= Rotate(alfa, 0,1,0)
TG1=TG1*AUX
modelMatrix(TG1)
pintaVaca()
TG2=TG1*TG2
modelMatrix(TG2)
pintaPatricio()
```

d)
```
AUX= Rotate(alfa, 0,1,0)
TG1=AUX*TG1
modelMatrix(TG1)
TG2=AUX*TG2
modelMatrix(TG2)
pintaVaca()
pintaPatricio()
```

Solució: a)

6. *(1 punt)* Tenim una piràmide de base quadrada de costat 5, amb la base centrada al punt (0,0,2.5) i alçada de la piràmide 5 amb l'eix en direcció Z+. A l'escena tenim també un cub de costat 5 centrat a l'origen. El viewport està definit amb `glViewport (0,0,400,800)`. Si a la vista es veu la imatge que teniu al dibuix (caseta), quines inicialitzacions d'una càmera axonomètrica (posició+orientació i òptica) permetrien veure aquesta imatge? Tots els angles estan en graus.



a)
```
PM=perspective (90, 1, 5, 10);
projectionMatrix (PM)
VM=translate (0,0,-10);
VM=VM*rotate (90,1,0,0);
VM=VM*translate (0,0,-2.5);
viewMatrix (VM);
pinta_escena ();
```

b)
```
PM=ortho (-2.5, 2.5, -5, 5, 5, 10);
projectionMatrix (PM)
VM=translate (0,0,-7.5);
VM=VM*rotate (90,0,0,1);
VM=VM*rotate (90,0,1,0);
VM=VM*translate (0,0,-2.5);
viewMatrix (VM);
pinta_escena ();
```

c)
```
PM=ortho (-2.5, 2.5, -5, 5, 5, 10);
projectionMatrix (PM)
VM=translate (0,0,-7.5);
VM=VM*rotate (-90,0,0,1);
VM=VM*rotate (90,0,1,0);
VM=VM*translate (0,0,-2.5);
viewMatrix (VM);
pinta_escena ();
```

d)
```
PM=ortho (-5, 5, -5, 5, 5, 10);
projectionMatrix (PM)
VM=translate (0,0,-7.5);
VM=VM*rotate (90,0,0,1);
VM=VM*rotate (90,0,1,0);
VM=VM*translate (0,0,-2.5);
viewMatrix (VM);
pinta_escena ();
```

Solució: b)

7. *(1 punt)* Es vol realitzar una vista en planta (visió des de dalt) d'una escena/objecte que està centrat a l'origen amb una capsa contenidora de mides 10x10x10. Quina de les següents definicions et sembla correcta per definir la posició + orientació de la càmera (per a calcular la viewMatrix)? Sabem que la càmera és perspectiva i els angles de les rotacions estan en graus.

   a) OBS = (0,10,0); VRP = (0,0,0); up = (0,1,0);
   VM = lookAt (OBS, VRP, up);
   viewMatrix(VM);

   b) OBS = (0,0,0); VRP = (0,10,0); up = (0,0,-1);
   VM = lookAt (OBS, VRP, up);
   viewMatrix(VM);

   c) VM = translate (0,0,-10);
   VM = VM * rotate (90, 1,0,0);
   viewMatrix(VM);

   d) VM = translate (0,0,-10);
   VM = VM * rotate (-90, 0,1,0);
   viewMatrix(VM);

   Solució: c)

8. *(1 punt)* Una aplicació permet, prement la tecla 'o', permutar entre una càmera perspectiva i una axonomètrica, ambdues amb el mateix ZNear i ZFar. Totes dues càmeres veuen l'escena completa i sense deformacions. Un estudiant dubte de quina càmera és la que està activa en un cert moment, quin dels següents experiments li aconsellaries fer per a deduir-ho?

   a) Movent l'observador en la direcció del VRP, la perspectiva retallarà per culpa del ZNear i l'axonomètrica no.

   b) Fent un resize de la finestra, la perspectiva deformarà i l'axonomètrica no.

   c) Movent l'observador en la direcció del VRP, la grandària de l'objecte no canviarà en l'axonomètrica, sí en la perspectiva.

   d) Girant la càmera en tercera persona (mitjançant angles d'Euler), l'axonomètrica retallarà i la perspectiva no.

   Solució: c)

Examen parcial d'IDI 2015-2016, Q1. 4/11/2015 12:30

Nom i cognoms: Temps: 1h 20'

3. *(1 punt)* Ordena de forma correcta els processos del pipeline de visualització projectiu d'OpenGL, és a dir, en quin ordre afecten aquests processos a la primitiva que s'envia a pintar:

   a) 1) ProjectTransform; 2) ViewTransform; 3) ModelTransform; 4) Retallat;
   b) 1) ModelTransform; 2) ViewTransform; 3) ProjectTransform; 4) Retallat;
   c) 1) ModelTransform; 2) ViewTransform; 3) Retallat; 4) ProjectTransform;
   d) 1) ViewTransform; 2) ModelTransform; 3) ProjectTransform; 4) Retallat;

   Solució: b)

4. *(1 punt)* Tenim una càmera axonomètrica definida amb els paràmetres: OBS = (5,0,0), VRP = (0,0,0), up = (0,1,0), Window = (-2,2,-2,2), Znear = 2, Zfar = 8. Indica quins paràmetres definirien el mateix volum de visió considerant que l'observador passa a estar en OBS = (0,5,0). La visió de la imatge final no té perquè ser la mateixa i la relació d'aspecte del viewport no és rellevant.

   a) VRP = (0,1,0), up = (0,0,1), Window = (-3,3,-2,2), Znear = 2, Zfar = 8.
   b) VRP = (0,0,0), up = (1,0,0), Window = (-2,2,-2,2), Znear = 2, Zfar = 8.
   c) VRP = (0,2,0), up = (0,0,1), Window = (-3,3,-2,2), Znear = 3, Zfar = 7.
   d) VRP = (0,0,0), up = (0,0,-1), Window = (-2,2,-2,2), Znear = 3, Zfar = 7.

   Solució: c)

5. *(1 punt)* Volem ubicar un model en una posició concreta d'una escena que es visualitza amb una càmera correctament definida. Tal i com indica el codi següent, hem passat al vèrtex shader com uniforms les matrius següents: TG que permet ubicar el model, View Matrix (VM) i Project Matrix (PM). Completa la instrucció que permet calcular les coordenades d'un vèrtex de l'objecte respecte el sistema de coordenades de l'observador (SCO).

```
in vec3 vertex;
uniform mat4 TG, VM, PM;

void main(){
   vec4 vobs;
   vobs =
    ...
}
```

   a) `vobs = TG*VM*PM*vec4(vertex,1.0);`
   b) `vobs = VM*TG*vec4(vertex,1.0);`
   c) `vobs = TG*VM*vec4(vertex,1.0);`
   d) `vobs = VM*vec4(vertex,1.0);`

   Solució: b)

6. *(1 punt)* Tenim un objecte centrat a l'origen i amb capsa contenidora de mides 3 d'ample, 3 d'alçada i 3 de profunditat. Es vol modificar **només** la seva alçada per a què passi a ser 2, quina de les següents TG és la correcta?

   a) `TG = glm::scale (glm::mat4(1.f), glm::vec3(1.0, 2.0, 1.0));`
   b) `TG = glm::scale (glm::mat4(1.f), glm::vec3(3.0, 2.0, 3.0));`
   c) `TG = glm::scale (glm::mat4(1.f), glm::vec3(1.0, 2.0/3.0, 1.0));`
   d) `TG = glm::scale (glm::mat4(1.f), glm::vec3(2.0/3.0, 2.0/3.0,2.0/3.0));`

   Solució: c)

7. *(1 punt)* Tenim una càmera en primera persona correctament definida en posicionament i en òptica. El viewport és de 500x500. Quina de les següents afirmacions és correcta respecte a la relació d'aspecte (`ra`)?

   a) S'ha de modificar `ra` sempre que és modifiqui el viewport sigui quin sigui el tipus de l'òptica.
   b) S'ha de modificar `ra` només si es modifica el viewport i l'òptica és perspectiva.
   c) S'ha de modificar `ra` si es modifica la finestra gràfica encara que el viewport no es modifiqui.
   d) En la càmera en primera persona mai s'ha de modificar la relació d'aspecte de la càmera.

   Solució: a)

8. *(1 punt)* Tenim una escena en la que utilitzem el codi següent per ubicar la càmera. Quins serien els paràmetres OBS, VRP i up que permetrien definir la mateixa càmera? (no modifiquem l'òptica).

```
VM = Translació(0,0,-10);
VM = VM*Rotacio_z (90);
VM = VM*Rotació_y (-90);
VM = VM*Translació (10,-10,0)
ViewMatrix (VM);
```

   a) OBS = (0,10,0), VRP = (10,10,0), up = (0,0,1)
   b) OBS = (10,10,0), VRP = (0,10,0), up = (0,1,0)
   c) OBS = (10,10,0), VRP = (-10,10,0), up = (0,0,1)
   d) OBS = (0,10,0), VRP = (-10,10,0), up = (0,0,-1)

   Solució: d)

9. *(1 punt)* Quina de les següents afirmacions **és incorrecta**?

   a) Si tenim una càmera axonomètrica i reduïm el seu window (respectant la seva relació d'aspecte), estem fent un zoom-in.
   b) Si incrementem el FOV de la càmera perspectiva, haurem d'incrementar la relació d'aspecte, per a què el window mantigui la seva proporció.
   c) L'algorisme de retallat (clipping) és el mateix sigui la càmera perspectiva o axonomètrica.
   d) L'eix Y del sistema de coordenades de l'observador (SCO) sempre es projecta vertical (direcció Y) en el sistema de coordenades de dispositiu (SCD).

   Solució: b)

Examen parcial d'IDI 2015-2016, Q2. 20/4/2016 12:30

Nom i cognoms: Temps: 1h 20'

3. *(1 punt)* Suposant que tenim la matriu de transformació de model (`TG`), la matriu de canvi de punt de vista (`view`) i la matriu de projecció (`proj`), quina és la multiplicació correcta que transforma el vèrtex (`vertex`) a coordenades de clipping? (el vèrtex ja està en coordenades homogènies, és a dir és un vec4(x,y,z,1)).

   a) `proj * TG * view * vertex`.
   b) `vertex * proj * view * TG`.
   c) `vertex * TG * view * proj`.
   d) `proj * view * TG * vertex`.

   Solució: d)

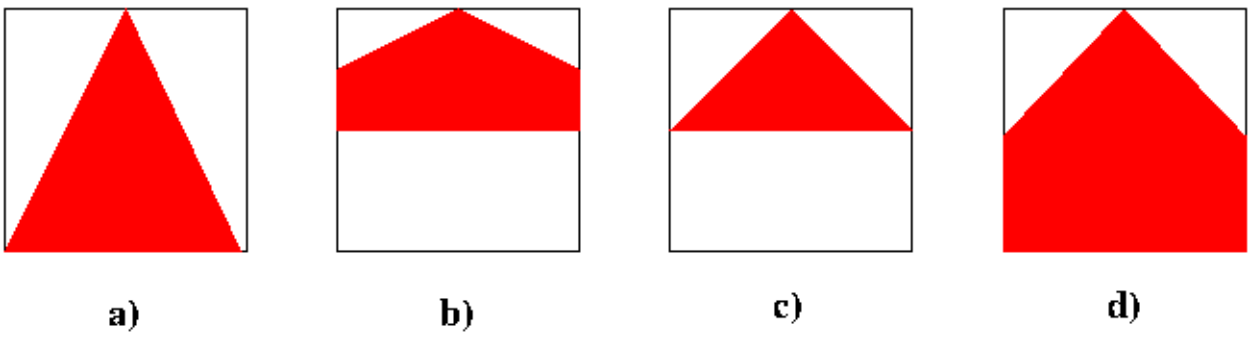
4. *(1 punt)* Un estudiant defineix la seva càmera amb OBS, d, v i up, essent d la distància entre OBS i VRP i v el vector normalitzat que va d'OBS a VRP, i calculant VRP com: **VRP = OBS + d * v**.

   En un cert moment, l'estudiant incrementa d i actualitza VRP i la view matrix però no la projection matrix, quin efecte tindrà en la visualització de l'escena?

   a) Com que no actualitza l'òptica, retallarà l'escena per Znear.
   b) Veurà l'escena més petita, el punt d'enfoc està més lluny.
   c) Veurà exactament el mateix.
   d) Afectarà en la deformació perspectiva que observarà.

   Solució: c)

5. *(1 punt)* Donada la descripció de l'escena de l'exercici 1 i havent inicialitzat les matrius de càmera (view) i projecció (proj) a la matriu identitat, indica quina de les següents imatges és la que sortirà en un viewport de 600x600 (sabem que el Vertex Shader i el Fragment Shader estan correctament implementats):



   Solució: b)

6. *(1 punt)* Imaginem que tenim l'escena de l'exercici 2, és a dir tenim les TGs necessàries per ubicar els dos objectes de forma correcta. Volem posicionar una càmera de manera que es vegi en el *viewport* el Patricio centrat i mirant cap a la càmera (de cara a la càmera). Quina d'aquestes inicialitzacions dels paràmetres de càmera seria correcta?

   a) VRP = (0, 6, 0), OBS = VRP + 2*radi_esfera*v, amb v=(0,1,0)), up = (1, 0, 0).

   b) VRP = ((Patxmin+Patxmax)/2, (Patymin+Patymax)/2, (Patzmin+Patzmax)/2), OBS = (VRP.x + 10, VRP.y, VRP.z), up = (0, 1, 0).

   c) VRP = (1, 6, 0), OBS = (15, 6, 0), up = (0, 1, 0).

   d) VRP = (0, 6, 0), OBS = (0, 6, 10), up = (0, 1, 0).

   Solució: c)

7. *(1 punt)* Tenim un objecte al que se li ha aplicat la següent transformació de model (TG) per a ubicar-lo a l'escena (C és el centre de l'objecte):

```
TG = Translació (3,0,3);
TG = TG*Rotacio_z (-90);
TG = TG*Rotacio_x (90);
TG = TG*Rotacio_y (90);
TG = TG*Escala (1/mida,1/mida,1/mida);
TG = TG*Translació (-C.x,-C.y,-C.z);
```

   Indica amb quina de les següents transformacions aconseguiríem l'objecte centrat al mateix punt i orientat de la mateixa manera però essent el doble de gran.

   a)
```
TG = Translació (3,0,3);
TG = TG*Rotacio_x (90);
TG = TG*Escala (2/mida,2/mida,2/mida);
TG = TG*Translació (-C.x,-C.y,-C.z);
```

   b)
```
TG = Translació (-3,0,-3);
TG = TG*Rotacio_z (90);
TG = TG*Rotacio_x (90);
TG = TG*Rotacio_y (-90);
TG = TG*Escala (2/mida,2/mida,2/mida);
TG = TG*Translació (-C.x,-C.y,-C.z);
```

   c)
```
TG = Translació (3,0,3);
TG = TG*Rotacio_z (-90);
TG = TG*Rotacio_x (90);
TG = TG*Rotacio_y (90);
TG = TG*Translació (-C.x,-C.y,-C.z);
TG = TG*Escala (2/mida,2/mida,2/mida);
```

   d) Cap de les altres és correcte.

   Solució: a)

Examen parcial d'IDI 2015-2016, Q2. 20/4/2016 12:30

Nom i cognoms: Temps: 1h 20'

8. *(1 punt)* Tenim una escena en la que utilitzem una càmera amb OBS = (-5,0,3), VRP = (5,0,3) i up = (0,0,1). Quin conjunt de transformacions geomètriques permetrien definir la mateixa càmera, és a dir, generar la mateixa viewMatrix? (no modifiquem l'òptica).

a)
```
VM = Translació(0,0,-10);
VM = VM*Rotacio_z (90);
VM = VM*Rotació_y (-90);
VM = VM*Translació (-5,0,-3)
ViewMatrix (VM);
```

b)
```
VM = Translació(0,0,-10);
VM = VM*Rotacio_z (90);
VM = VM*Rotació_y (90);
VM = VM*Translació (-5,0,-3)
ViewMatrix (VM);
```

c)
```
VM = Translació (-5,0,-3)
VM = VM*Rotació_y (90);
VM = VM*Rotacio_z (90);
VM = VM*Translació(0,0,-10);
ViewMatrix (VM);
```

d)
```
VM = Translació (-5,0,-3)
VM = VM*Rotació_y (-90);
VM = VM*Rotacio_z (90);
VM = VM*Translació(0,0,-10);
ViewMatrix (VM);
```

Solució: b)

9. *(1 punt)* La crida `TP = perspective (M_PI/4, 1, 3, 6)`, defineix la matriu de projecció d'una càmera perspectiva. La ra del *viewport* és 1. S'envia a pintar un cub d'aresta 2 centrat a l'origen i es veu sencer com un quadrat que ocupa tot el *viewport*. Si es modifica la ra del window i passa a ser 2 (i no es modifica cap altre paràmetre de la càmera), què es veurà quan tornem a pintar el cub?

a) Veurem un rectangle el doble d'alt que d'ample.

b) Com que ra del window és més gran que 1 es segueix veient un quadrat.

c) Veurem un rectangle el doble d'ample que d'alt.

d) Si no modifiquem el FOV el quadrat quedarà retallat.

Solució: a)

2. *(1 punt)* Pensant en l'escena de l'exercici 1, quins dels següents paràmetres d'una càmera axonomètrica serien adients per a poder veure, en la posició inicial dels Patricios i en un viewport quadrat, únicament el Patricio de dalt del pilar (Pat2) i de manera que aquest estigui dret i de cara a la càmera?

   a) OBS = (5,4,5); VRP = (5,4,0); up = (0,1,0);
   left = -2; right = 2; bottom = -2; top = 2; Znear = 2; Zfar = 8;
   b) OBS = (5,4,-5); VRP = (5,4,0); up = (0,1,0);
   left = -2; right = 2; bottom = -2; top = 2; Znear = 2; Zfar = 8;
   c) OBS = (5,4,-5); VRP = (5,4,1); up = (0,1,0);
   left = -1; right = 1; bottom = -1; top = 1; Znear = 2; Zfar = 8;
   d) OBS = (5,4,-5); VRP = (5,4,0); up = (1,0,0);
   left = -1; right = 1; bottom = -1; top = 1; Znear = 3; Zfar = 7;

   Solució: c)

3. *(1 punt)* Tenint en compte l'escena dels dos Patricios en la seva posició inicial (escena de l'exercici 1) i una càmera posicionada i orientada amb el VRP al centre de l'escena i l'observador a distància $d = 6.0$ del VRP en una certa direcció, indica quins paràmetres d'una òptica perspectiva serien adients per a veure l'escena (és a dir el pilar dels dos Patricios) sencera, sense deformar i optimitzant el viewport amb una càmera en tercera persona (per inspecció). Considereu que el radi de l'esfera contenidora és $R = 3.0$ i que el viewport és de 600x400 píxels.

   a) raw = 1.5; FOV = atan(tan(R/2)/raw); ZNear = 3; ZFar = 9;
   b) FOV = 2*asin(R/6); raw = 1.5; ZNear = 3; ZFar = 9;
   c) FOV = 2*asin(R/6); raw = 1; ZNear = 3; ZFar = 9;
   d) FOV = 2*atan(R/6); raw = 1.5; ZNear = 6; ZFar = 9;

   Solució: b)

4. *(0.5 punts)* Dos estudiants discuteixen respecte a la implementació del zoom amb òptica axonomètrica (ortogonal) i perspectiva. Quina de les seves afirmacions és certa?

   a) En òptica ortogonal només es pot obtenir un efecte de zoom modificant OBS i VRP en la direcció de visió.
   b) En òptica perspectiva cal modificar FOV, Znear i Zfar.
   c) En les dues òptiques es pot fer zoom modificant el window de la càmera.
   d) En òptica perspectica si avancem OBS i VRP en la direcció de visió cal anar amb compte amb la ra.

   Solució: c)

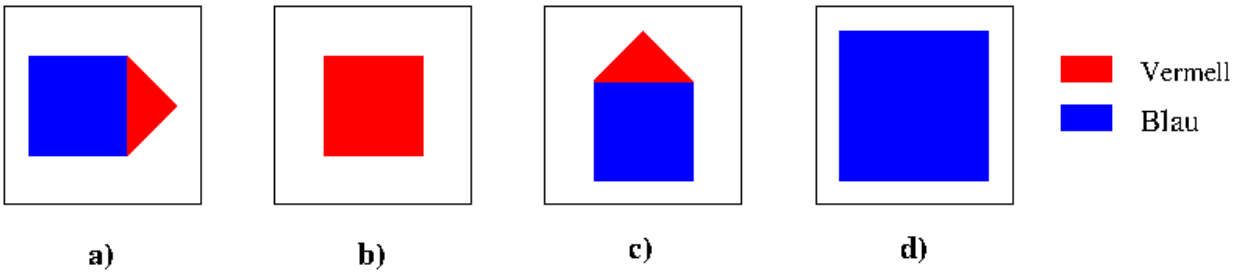
Examen parcial d'IDI 2016-2017, Q1. 26/10/2016 12:30

Nom i cognoms: Temps: 1h 45'

5. *(1 punt)* Tenim una escena amb una caseta formada per:

   - un cub de color blau de costat 20 amb les cares paral·leles als plans coordenats i amb el centre de la seva cara inferior situat en el punt (10,0,0).
   - una piràmide de color vermell de base quadrada d'aresta 20 i alçada 10, ubicada just a sobre del cub, amb la base de la piràmide coincidint amb la cara superior del cub.

   Al pintar la caseta en un viewport quadrat amb els paràmetres de càmera: OBS = (10,40,0); VRP = (10,-30,0); up = (1,0,0); FOV = 90 (graus); ra = 1.0; Znear = 10; Zfar = 45; Quina de les següents figures representa la imatge resultant?



   Solució: b)

6. *(1 punt)* Quin dels següents codis per a inicialitzar la *viewMatrix* generaria la mateixa matriu que la càmera descrita a la pregunta 5?

   a)
   ```
   VM = Translació(0,0,-40);
   VM = VM*Rotacio_z (90);
   VM = VM*Rotació_x (90);
   VM = VM*Translació (-10,0,0)
   viewMatrix (VM);
   ```

   b)
   ```
   VM = Translació(0,0,-40);
   VM = VM*Rotacio_z (90);
   VM = VM*Rotació_y (90);
   VM = VM*Translació (-10,0,0)
   viewMatrix (VM);
   ```

   c)
   ```
   VM = Translació (-10,30,0)
   VM = VM*Rotació_x (90);
   VM = VM*Rotacio_z (90);
   VM = VM*Translació(0,0,-70);
   viewMatrix (VM);
   ```

   d)
   ```
   VM = Translació (-10,0,0)
   VM = VM*Rotació_y (-90);
   VM = VM*Rotacio_z (90);
   VM = VM*Translació(0,0,-40);
   viewMatrix (VM);
   ```

   Solució: a)

7. *(0.5 punts)* Un estudiant implementa el Vertex Shader oblidant-se de multiplicar els vèrtexs per la *viewMatrix*, és a dir, el càlcul del `gl_Position` el fa fent:

   ```
   gl_Position = proj * TG * vec4(vertex, 1.0);
   ```

   on `TG` és la *modelMatrix* i `proj` és la *projectMatrix* d'una òptica perspectiva. Quina afirmació és la correcta?

   a) El volum de visió queda definit pels punts (-1,-1,-1) i (1,1,1) en coordenades de l'aplicació (SCA).
   b) Té el mateix efecte que tenir OBS = (0,0,0), VRP = (0,-10,0) i up = (0,1,0).
   c) No podem conèixer la posició de la càmera.
   d) Cap de les altres respostes és correcta.

   Solució: d)

8. *(1 punt)* Tenim una escena amb un terra i un homer que es troba damunt del terra i volem moure el homer sobre el terra com si caminés. Tenint en compte que totes les matrius de TG es calculen i envien dins del `pintaEscena()`:

   a) Caldrà declarar dos uniforms per a les dues matrius de TG en el Vertex Shader (TG1 i TG2), i mutiplicar per una o l'altra depenent del VAO que estem pintant.

   b) Si tenim un mètode `modelTransform()` que calcula la matriu de translació del homer i l'envia al Vertex Shader, el pseudocodi del `pintaEscena()` seria:

```
pintaEscena() {
    modelTransform();
    pintaHomer();
    pintaTerra();
}
```

   c) Si tenim un únic mètode `modelTransform()` que calcula la matriu de translació del homer i l'envia al Vertex Shader, el pseudocodi del `pintaEscena()` seria:

```
pintaEscena () {
    pintaTerra();
    modelTransform();
    pintaHomer();
}
```

   d) Cap de les altres respostes és correcta.

   Solució: d)

9. *(0.5 punts)* Quin efecte tindría en la imatge resultant si en la càmera descrita en l'exercici 5 canviéssim el vector up i poséssim up = (0,0,1)?

   a) Sempre que modifiquem el vector up cal també modificar OBS.
   b) Per poder veure la caseta hauríem de modificar també l'òptica de la càmera.
   c) Veuríem una cara lateral del cub.
   d) La imatge resultant seria idèntica a la de la pregunta 5.

   Solució: d)

10. *(0.5 punts)* En l'escena de la caseta de la pregunta 5, en enviar-la a pintar amb una altra càmera, es veu la caseta allargada, és a dir, el cub, per exemple, es veu com un prisma amb la base el doble d'amplada que l'alçada. Què és el que ho provoca?

    a) La ra del viewport (rav) és < 1 i no s'ha modificat el FOV.
    b) La ra del window (raw) i la del viewport (rav) no són iguals.
    c) La ra del window (raw) és > 1 i la del viewport (rav) és 1.
    d) La ra del window (raw) és > 1 i la del viewport (rav) és > 1.

    Solució: b)

11. *(0.5 punts)* Per a què el procés de visualització funcioni correctament pel que respecta a la projecció de la geometria de l'escena, hem de programar obligatòriament els VS (Vertex Shader) i FS (Fragment Shader) de manera que:

    a) En el VS es calculin les coordenades de clipping del vèrtex i en el FS les de dipositiu.
    b) En el VS es calculin les coordenades d'observador del vèrtex i en el FS les de clipping.
    c) En el VS es calculin les coordenades de clipping del vèrtex.
    d) En el VS es calculin les coordenades de clipping i en el FS les d'observador.

    Solució: c)

Examen parcial d'IDI 2017-2018, Q1. (Perm: A) 8/11/2017 12:15

Nom i cognoms:

Normativa del test

(a) A les graelles que hi ha a continuació, marca amb una creu les teves respostes de l'examen. **No es tindrà en compte cap resposta fora d'aquestes graelles**.

(b) No es poden usar apunts, calculadores ni cap dispositiu electrònic.

(c) Totes les preguntes tenen una única resposta correcta.

(d) Les preguntes contestades de forma errònia tenen una **penalització del 33%**.

Num	A	B	C	D
3	**X**			
4		**X**		
5	**X**			
6		**X**		
7			**X**	

Num	A	B	C	D
8			**X**	
9				**X**
10			**X**	
11	**X**			
12			**X**	

3. *(1 punt)* Quins dels següents paràmetres de posició i orientació de la càmera permetrien veure l'escena de l'exercici 2 en forma de lletra C?

   a) OBS = (0,6,6); VRP = (0,6,0); up = (1,0,0).
   b) OBS = (0,6,5); VRP = (0,6,0); up = (0,1,0).
   c) OBS = (0,6,-5); VRP = (0,0,0); up = (1,0,0).
   d) Cap de les altres respostes és correcta.

4. *(1 punt)* Suposem una càmera amb paràmetres OBS=(10,6,0); VRP=(0,6,0); i up=(0,0,-1) per a veure l'escena de l'exercici 2. Quina de les següents transformacions geomètriques aconsegueix la mateixa View Matrix (VM) que aconseguiria la crida lookAt(OBS,VRP,up)?

   a) VM = T(0,0,-10) * Rz(90) * Rx(90) * T(0,-6,0)
   b) VM = T(0,0,-10) * Rz(90) * Ry(-90) * T(0,-6,0)
   c) VM = T(0,-6,0) * Ry(-90) * Rz(90) * T(0,0,-10)
   d) VM = T(0,-6,0) * Ry(-90) * T(0,0,-10)

5. *(1 punt)* Tenim un cub de costat 2 amb vèrtex de coordenades mínimes a l'origen de coordenades. Quina de les següents seqüències d'inicialització de posició de càmera permetrien (suposant que l'òptica és ortogonal i és correcta) veure en pantalla un hexàgon? (Nota: Considera els angles en graus)

   a)
   ```
   OBS=(3,3,3); VRP=(0,0,0); up=(0,1,0);
   VM = lookAt (OBS, VRP, up);
   viewMatrix(VM);
   ```

   b)
   ```
   OBS=(2,2,2); VRP=(0,0,0); up=(1,1,1);
   VM = lookAt (OBS, VRP, up);
   viewMatrix(VM);
   ```

   c)
   ```
   VM = Translate (0,0,-7);
   VM = VM * Rotate_Y (45);
   VM = VM * Translate (-1,-1,-1);
   viewMatrix(VM);
   ```

   d)
   ```
   VM = Translate (0,0,-3);
   VM = VM * Rotate_Z (45);
   VM = VM * Rotate_Y (-45);
   VM = VM * Translate (-1,-1,-1);
   viewMatrix(VM);
   ```

6. *(1 punt)* Donada l'escena del cub de l'exercici 5 i amb una càmera ortogonal que permet veure en pantalla un hexàgon sencer centrat al viewport, quin efecte tindria en la imatge si a aquesta càmera li anem modificant el vector d'up i actualitzant la matriu de VM corresponentment?

   a) Haurem de modificar la raw per continuar veient tot l'hexàgon sense retallar.
   b) Es veurà l'hexàgon igual que es veia però girat respecte del seu centre.
   c) No ho podem saber si no coneixem exactament l'òptica de la càmera.
   d) En comptes d'un hexàgon veurem un rectangle.

7. *(0.5 punts)* Indica quina de les següents associacions relaciona correctament cada matriu de transformació de Sistema de Coordenades amb el canvi de Sistemes de Coordenades que realitza. Recorda que els sistemes de coordenades són: SCM: Sistema de Coordenades de Model; SCA: Sistema de Coordenades d'Aplicació; SCO: Sistema de Coordenades d'Observador; SCC: Sistema de Coordenades de Clipping.

   a) VM: SCM $\rightarrow$ SCO; PM: SCO $\rightarrow$ SCC; TG: SCA $\rightarrow$ SCM
   b) PM: SCO $\rightarrow$ SCC; TG: SCM $\rightarrow$ SCO; VM: SCA $\rightarrow$ SCM
   c) PM: SCO $\rightarrow$ SCC; TG: SCM $\rightarrow$ SCA; VM: SCA $\rightarrow$ SCO
   d) PM: SCA $\rightarrow$ SCO; TG: SCM $\rightarrow$ SCA; VM: SCO $\rightarrow$ SCC

8. *(0,5 punts)* Donat el següent codi en el Vertex Shader, i on VM és la view matrix, podem afirmar que:

```
gl_Position = VM * vec4 (vertex,1);
```

   a) Que el punt (0,0,0) de SCA està dins del Volum de Visió
   b) Que el retallat no funcionarà perquè no estem multiplicant el vèrtex per cap matriu de projecció.
   c) Que el volum de visió és un cub que en SCO va entre (-1,-1,-1) i (1,1,1)
   d) Que el volum de visió és un cub que en SCA va entre (-1,-1,-1) i (1,1,1)

9. *(0.5 punts)* Per a què el procés de visualització funcioni correctament, cal que:

   a) En el Vertex Shader calculem les coordenades de clipping de cada vèrtex i en el Fragment Shader calculem les coordenades de dispositiu i el color.
   b) En el Vertex Shader calculem les coordenades de dispositiu de cada vèrtex i en el Fragment Shader assignem color als fragments.
   c) Com a sortida del Vertex Shader tinguem el color del vèrtex i en el Fragment Shader calculem les coordenades del fragment.
   d) Cap de les altres respostes és correcta.

10. *(0.5 punts)* Donada una view Matrix (VM) i un Viewport. Què cal assegurar en els paràmetres de la project Matrix (PM) per a evitar que l'escena que es pinta quedi deformada?

    a) Si rav < 1 cal igualar raw amb rav (raw = rav), sinó no.
    b) Només cal augmentar el fov.
    c) Només cal igualar raw amb rav (raw = rav).
    d) No hi ha prou amb igualar raw amb rav (raw = rav), cal també modificar el fov.

11. *(0.5 punts)* Es vol definir una càmera perspectiva per veure una escena en tercera persona. Hem ubicat VRP al centre de l'esfera contenidora de l'escena. L'esfera té radi R. L'observador està fora de l'esfera a una distància d del VRP. Sabem que el viewport és quadrat. Per calcular el FOV d'una càmera perspectiva que permeti veure l'esfera sencera digues quina afirmació és certa:

    a) Ho podem fer independentment del valor de Znear.
    b) Cal conèixer Znear i Zfar.
    c) Si coneixem Znear no ens cal el valor de d.
    d) Cap de les altres respostes és correcta.

12. *(0.5 punts)* Què es pot dir d'un color que en HSB (o HSV) té un valor màxim en la component S?

    a) Que és un color gris i la seva intensitat ve donada pel valor de B (o V).
    b) Que és un color pur i per tant com a mínim un dels components del model RGB ha de ser 1.
    c) Que és un color pur i per tant com a mínim un dels components del model RGB ha de ser 0.
    d) No en podem dir res sense saber els valors de H i de B (o V).

Examen parcial d'IDI 2018-2019, Q1. (Perm: A) 5/11/2018 12:00

Nom i cognoms:

Normativa del test

(a) A les graelles que hi ha a continuació, marca amb una creu les teves respostes de l'examen. **No es tindrà en compte cap resposta fora d'aquestes graelles**.

(b) No es poden usar apunts, calculadores ni cap dispositiu electrònic.

(c) Totes les preguntes tenen una única resposta correcta.

(d) Les preguntes contestades de forma errònia tenen una **penalització del 33%** del valor de la pregunta.

Num	A	B	C	D
7				**X**
8			**X**	
9				**X**

Num	A	B	C	D
10	**X**			
11			**X**	
12	**X**			

Num	A	B	C	D
13	**X**			
14				**X**
15			**X**	

7. *(0.5 punts)* Donada una càmera perspectiva en primera persona, un estudiant implementa la funcionalitat d'avançar l'observador en la direcció de visió i observa que després d'haver avançat uns quants cops, l'observador mira en sentit contrari al qual avança, la qual cosa és clarament un error.

   a) Això no pot passar, deu tenir un error en la definició de l'escena.

   b) Deu haver fet un resize i no ha modificat la relació d'aspecte (ra) del window apropiadament i li retalla.

   c) Deu haver modificat la inclinació (up) de la càmera (tot mantenint la direcció de visió).

   d) Deu haver modificat OBS per avançar però no VRP.

8. *(0.5 punts)* Indica quina de les següents inicialitzacions permet obtenir una visualització en planta (projecció en el pla X-Z de l'aplicació) d'una escena de la qual sabem que la seva esfera contenidora està centrada a l'origen i té radi 20.

   a) VM=I ; VM= VM*Gx(90); VM=VM*Translacio (0,0,-30)

   b) VM=I ; VM=VM*Translacio (0,0,-30); VM= VM*Gy(90);

   c) OBS= (0,50,0), VRP=(0,10,0), up=(0,0,1)

   d) OBS=(0,50,0), VRP= (0,0,0), up=(0,1,0)

9. *(0.5 punts)* Definint la viewMatrix utilitzant transformacions geomètriques a partir dels angles d'Euler:

   a) No es pot reproduir l'efecte de modificar el vector up.

   b) L'òptica obligatòriament ha de ser perspectiva.

   c) Aquesta matriu no podrà ser la identitat.

   d) Podem obtenir la mateixa visualització de l'escena que si s'ha obtingut a partir de lookAt.

10. *(0.5 punts)* En el procés de visualització, l'algorisme de retallat implementat per OpenGL (clipping) per a triangles que es pintaran omplerts de color...

    a) és el mateix algorisme tant si la càmera és perspectiva com si és ortogonal.
    b) és funció de si el tipus de càmera és perspectiva o és ortogonal.
    c) és funció de si el volum de visió és un tros de piràmide o un paral·lelepípede.
    d) és funció de si les coordenades dels vèrtexs a la sortida del vertex shader (gl_Position) estan en coordenades de clipping o no.

11. *(0.5 punts)* Tenim una funció pinta_cub que envia a pintar un cub d'aresta 1 centrat a l'origen de coordenades i una càmera ortogonal que permet visualitzar-ho correctament. Volem que l'usuari, mitjançant una tecla, pugui decidir si vol que el cub es vegi escalat al doble de gran o no. Com ho implementem?

    a) Podem enviar un uniform al FS (amb valors 1 o 2) i, en cas que tingui valor 2, desplaçar els fragments en el viewport per ocupar més espai.
    b) Sense modificar la posició de la càmera no es pot aconseguir cap escalat.
    c) Podem enviar un uniform al VS (amb valors 1 o 2) i que es multipliquin les coordenades dels vèrtexs pel uniform.
    d) Es pot aconseguir aquest efecte reprogramant tant el VS com el FS indistintament.

12. *(0,5 punts)* Es vol modificar una aplicació per a què cada crida a paintGL pinti la mateixa escena en dos viewports diferents. L'escena es pinta mitjançant una crida pinta_escena(). Els dos viewports defineixen respectivament el quadrant inferior esquerre i el quadrant superior dret de la finestra gràfica, que és de 800x600 píxels, tots dos tenen la mateixa relació d'aspecte. La posició i orientació de la càmera no es volen modificar. Per a fer això caldrà fer canvis a paintGL:

    a) Hem d'afegir una segona crida a pinta_escena(), precedint cada crida a pinta_escena() de la definició del viewport corresponent.
    b) Tant sols caldrà modificar la definició del viewport, la raw i la projectMatrix abans d'una segona crida a pinta_escena() que afegirem.
    c) Tant sols caldrà fer dos crides a glViewport, sense necessitat d'afegir res més.
    d) Afegirem una segona crida a pinta_escena(), però redefinirem la projectMatrix abans de cadascuna de les dues crides a aquesta funció.

13. *(0.5 punts)* Un objecte està modelat per un conjunt de triangles:

    a) L'esquema de representació del model pot indicar la topologia (ordre d'unió dels vèrtex) de manera implícita o explícita.
    b) L'esquema de representació ha de ser sempre una única taula de vèrtexs sense repetició.
    c) Per intercanviar la informació amb OpenGL utilitzant VBOs (com fem al laboratori) cal que no hi hagi repetició de vèrtexs i que la topologia (ordre d'unió dels vèrtex) sigui explícita.
    d) L'esquema de representació ha d'indicar sempre la topologia (ordre d'unió dels vèrtex) de manera implícita.

14. *(0.5 punts)* Els valors per defecte en un formulari...

    a) s'han de posar sempre que hi hagi un percentatge superior al 30% de probabilitat de que sigui el correcte.
    b) són de poca utilitat però els usuaris s'hi han acostumat.
    c) si es posen, cal posar-los per tots els camps.
    d) no necessàriament calen per a tots els camps.

15. *(0.5 punts)* Tenint en compte que defineix una aplicació per indicar el temps meteorològic, quin tipus de representació es correspon a la icona de la imatge?

    a) Similaritat
    b) Simbòlic
    c) Exemple
    d) Arbitrari

Examen parcial d'IDI 2018-2019, Q2. (Perm: A) 8/4/2019 14:00

Nom i cognoms:

Normativa del test

(a) A les graelles que hi ha a continuació, marca amb una creu les teves respostes de l'examen. **No es tindrà en compte cap resposta fora d'aquestes graelles**.

(b) No es poden usar apunts, calculadores ni cap dispositiu electrònic.

(c) Totes les preguntes tenen una única resposta correcta.

(d) Les preguntes contestades de forma errònia tenen una **penalització del 33%** del valor de la pregunta.

Num	A	B	C	D
9		**X**		
10	**X**			
11				**X**
12	**X**			

Num	A	B	C	D
13	**X**			
14		**X**		
15	**X**			
16		**X**		

9. *(0.5 punts)* Tenint en compte la descripció de l'escena 1 feta en el full 1 de l'examen i considerant com a posicionament de càmera: OBS=(15,2,-6); VRP=(0,2,-6); i up=(0,1,0) (ja vista en un exercici anterior). Què es veuria en el viewport si tenim una òptica ortogonal amb znear=1, zfar=30 i window=(-3,3,-3,3)? El viewport és quadrat.

a) Veuríem els tres Patricios en profunditat i el terra.

b) Veuríem un únic Patricio i no veuríem el terra.

c) Veuríem el legoman davant dels tres Patricios i el terra.

d) Veuríem un únic Patricio i un legoman, no veuríem el terra.

10. *(0.5 punts)* El procés que obté les coordenades de dispositiu d'un vèrtex dins del procés de visualització:

a) Només requereix el viewport i les coordenades normalitzades del vèrtex.

b) Requereix el vèrtex en coordenades de clipping, el window de la càmera i el viewport.

c) Les dades que requereix depenen de si es realitza abans o després del Fragment Shader.

d) Podem triar si es realitza abans o després del clipping.

11. *(0.5 punts)* Quan definim la posició i orientació de la càmera (viewMatrix) mitjançant angles d'Euler:

a) No es pot emular l'efecte del vector up que tenim quan la definim amb lookAt.

b) Sempre és equivalent a tenir un up=(0,1,0).

c) Cal utilitzar lookAt per ubicar la càmera i lookAt seguit de transformacions d'Euler quan tenim interacció.

d) Modificant l'angle d'Euler de gir respecte Z podem obtenir l'efecte del vector up.

12. *(0.5 punts)* Sabem que a una impressora CMY li falla la tinta magenta (però continua imprimint). Si en imprimir un dibuix en aquesta impressora, en el dibuix imprès es veuen dues franges (F1 i F2) de colors F1 = cian i F2 = groc, de quins colors són les franges del dibuix original?

    a) F1 = blau i F2 = groc
    b) F1 = vermell i F2 = groc
    c) F1 = blau i F2 = blau
    d) F1 = vermell i F2 = cian

13. *(0.5 punts)* Es disposa d'una funció pintacub() que envia a pintar el VAO d'un cub d'aresta 10 amb el centre del cub a l'origen de coordenades. Tenim una càmera definida amb OBS= (0,0,20), VRP=(0,0,0), up=(0,1,0) Quin dels següents codis és correcte per visualitzar una escena formant una L amb el cub a la cantonada de la L centrat a l'origen de coordenades. Nota: el vertex shader està correctament definit i rep com uniform una matriu TG de 4x4; l'òptica és ortogonal i està definida també correctament.

    a)
    ```
    TG=I; TG= TG*Translació (0,0,10); modelMatrix (TG)
    Pintacub()
    TG=I; TG= TG*Translació (0,10,0); modelMatrix (TG)
    Pintacub()
    TG=I; TG= TG*Translació (10,0,0); modelMatrix (TG)
    Pintacub()
    ```
    b)
    ```
    TG1=I; TG1= TG1*Translació (0,0,10); modelMatrix (TG1)
    TG2=I; TG2= TG2*Translació (0,10,0); modelMatrix (TG2)
    TG3=I; TG3= TG3*Translació (10,0,0); modelMatrix (TG3)
    Pintacub()
    Pintacub()
    Pintacub()
    ```
    c)
    ```
    Pintacub()
    TG=I; TG= TG*Translació (0,0,10); modelMatrix (TG)
    Pintacub()
    TG=I; TG= TG*Translació (0,10,0); modelMatrix (TG)
    Pintacub()
    TG=I; TG= TG*Translació (10,0,0); modelMatrix (TG)
    ```
    d)
    ```
    TG=I; TG= TG*Translació (0,0,10); modelMatrix (TG)
    Pintacub()
    TG= TG*Translació (0,10,0); modelMatrix (TG)
    Pintacub()
    TG= TG*Translació (10,0,0); modelMatrix (TG)
    Pintacub()
    ```

14. *(0.5 punts)* Aquesta indicació de progrés...

    a) No indica res, no dóna cap informació.
    b) No és útil si la feina a fer triga més de 15 segons.
    c) Serveix sempre perquè indica que està treballant.
    d) Cap de les anteriors.

15. *(0.5 punts)* Tenint en compte que defineix una aplicació per indicar l'hora, quin tipus de representació es correspon a la icona de la imatge?

a) Similaritat

b) Simbòlic

c) Exemple

d) Arbitrari

16. *(0.5 punts)* Del següent llistat d'afirmacions relatius a l'us de colors en una interfície gràfica, indica quina és FALSA:

a) El colors saturats poden provocar fatiga visual.

b) Es millor seleccionar molts colors diferents per a les icones d'una interfície, per a que sembli més senzill distingir els elements.

c) Per a que es vegi bé un text, es recomanable utilitzar contrast, per exemple obscur per al text, i suau per al fons.

d) Si tenim botons del mateix color, sobre un fons que canvia d'intensitat, l'usuari els veurà diferents.