

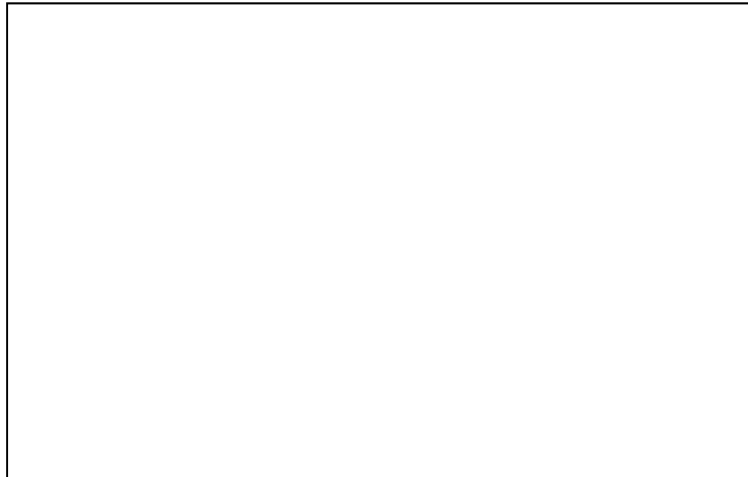
Nom:

---

**Exercici 1**

Copia a la dreta aquestes quatre tasques del pipeline gràfic, però ordenades d'acord amb l'ordre d'execució.

- Alpha Test
- glDrawElements
- Rasterization
- Vertex Shader

**Exercici 2**

Copia a la dreta aquestes quatre tasques del pipeline gràfic, però ordenades d'acord amb l'ordre d'execució.

- Depth test
- Geometry Shader
- Rasterization
- setUniformValue

**Exercici 3**

Escriu quin és l'espai de coordenades inicial i final de la multiplicació de la **projection matrix** per un vèrtex.

Inicial:

Final:

#### Exercici 4

Quina mena de punt o vector estem transformant amb el producte que apareix a sota?

$$\begin{bmatrix} x_a \\ y_a \\ z_a \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}^{-T} \begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix}$$

#### Exercici 5

Escriu, per cada tasca, si s'executa ABANS o DESPRÉS del FS:

- (a) Pas a NDC
- (b) Geometry Shader
- (c) Divisió de perspectiva
- (d) Depth Test

#### Exercici 6

Indica, per cada punt, si pot ser dins (DINS) o segur que és fora (FORA) de la piràmide de visió d'una càmera perspectiva:

- (a) (0.2, -0.5, 0.8, 1) en clip space
- (b) (0, 0, 1, 1) en eye space
- (c) (200, 300, 400) en NDC
- (d) (0, 0, 0) en NDC

#### Exercici 7

Completa, en GLSL, una possible definició de la funció mix:

```
vec3 mix(vec3 a, vec3 b, float t)
{
    return ....
}
```

#### Exercici 8

Siguin R, S i T les matrius de rotació, escalat i translació a aplicar a un model com a part de la transformació de modelat. Indica en quin ordre és més habitual multiplicar aquestes matrius (escriu el producte de les matrius):

### Exercici 9

Escriu, usant la notació  $L(D|S)*E$ , els light paths que suporten aquestes tècniques:

(a) Two-pass raytracing

(b) Classic Raytracing

### Exercici 10



Volem aplicar la textura a un quad que té coordenades de textura inicials en  $[0,1]$ .

Completa el FS per aconseguir els resultats que es mostren:



(a)

frontColor = texture(colorMap, \_\_\_\_\_ \* vtexCoord);



(b)

frontColor = texture(colorMap, \_\_\_\_\_ \* vtexCoord);

### Exercici 11

Quin concepte de radiometria/fotometria és el més adient per mesurar la quantitat d'energia per unitat de temps que arriba a una superfície, per unitat d'àrea (unitats  $W/m^2$ )?

### Exercici 12

A l'equació general del rendering:

$$L_o(\mathbf{x}, \omega_o, \lambda, t) = L_e(\mathbf{x}, \omega_o, \lambda, t) + \int_{\Omega} f_r(\mathbf{x}, \omega_i, \omega_o, \lambda, t) L_i(\mathbf{x}, \omega_i, \lambda, t) (\omega_i \cdot \mathbf{n}) d\omega_i$$

Què representa  $\omega_o$ ?

### Exercici 13

Aquest VS calcula coordenades de textura per a un FS que implementa shadow mapping:

```
uniform mat4 lightMatrix;  
out vec4 textureCoords;  
const float a = .....;  
...  
void main() {  
    ...  
    textureCoords = T(a,a,a)*S(a,a,a)*lightMatrix*vec4(vertex,1);  
    gl_Position = modelViewProjectionMatrix *vec4(vertex,1);  
}
```

Usant aquesta notació:

$S(sx,sy,sz)$  -> Scale matrix

$T(tx,ty,tz)$  -> Translate matrix

$M$  -> model matrix (of the object)

$V$  -> view matrix (of the light camera)

$P$  -> projection matrix (of the light camera)

(a) Quin valor ha de tenir la constant  $a$ ?

(b) Escriu (com a producte de matrius) com l'aplicació ha de calcular, en aquest cas, la matriu **lightMatrix**.

### Exercici 14

Escriu la matriu o producte de matrius per les conversions següents, usant la notació:

$M$  = modelMatrix

$M^{-1}$  = modelMatrixInverse

$V$  = viewingMatrix

$V^{-1}$  = viewingMatrixInverse

$P$  = projectionMatrix

$P^{-1}$  = projectionMatrixInverse

$N$  = normalMatrix

$I$  = Identitat

a) Convertir un vèrtex de world space a clip space

b) Convertir un vèrtex de eye space a world space

### Exercici 15

Sigui  $P(u,v)$  la representació paramètrica d'una superfície, amb normals unitàries  $N(u,v)$ . Sigui  $F(u,v)$  un mapa d'elevacions. Escriu l'expressió que permet calcular la superfície  $P'(u,v)$  que resulta de pertorbar  $P$  segons el mapa d'elevacions, tal i com es faria servir en *displacement mapping*.

### Exercici 16

Escriu en codi GLSL com calcular la posició de la càmera en *object space*. Pot usar les matrius per defecte del viewer.

vec3 obs =

### Exercici 17

Un VS ha calculat unes coordenades de textura *vtexCoord* usant la tècnica de **projective texture mapping**.

(a) Indica quina ha de ser la declaració (en GLSL) de *vtexCoord*, al VS

(b) Completeu com caldria accedir a la textura *colormap* en aquest cas:

vec4 color = texture(colormap, .....);

### Exercici 18

En la tècnica de generació d'ombres per projecció,

(a) Quants cop cal dibuixar l'objecte que produeix l'ombra?

(b) Per què ens pot ser útil usar el stencil buffer?

### Exercici 19

En quin sistema de coordenades han d'estar aquestes variables per tal que el VS sigui correcte?

(a) `vec3 L = normalize(lightPosition.xyz - modelViewMatrix * P);` // L is the light vector

P ha d'estar en espai \_\_\_\_\_

(b) `gl_Position = projectionMatrix * P;`

P ha d'estar en espai \_\_\_\_\_

### Exercici 20

Si fem servir el mode de filtrat `GL_LINEAR_MIPMAP_LINEAR` per MINIFICATION, quants texels es fan servir per avaluar cada crida a `texture()`?