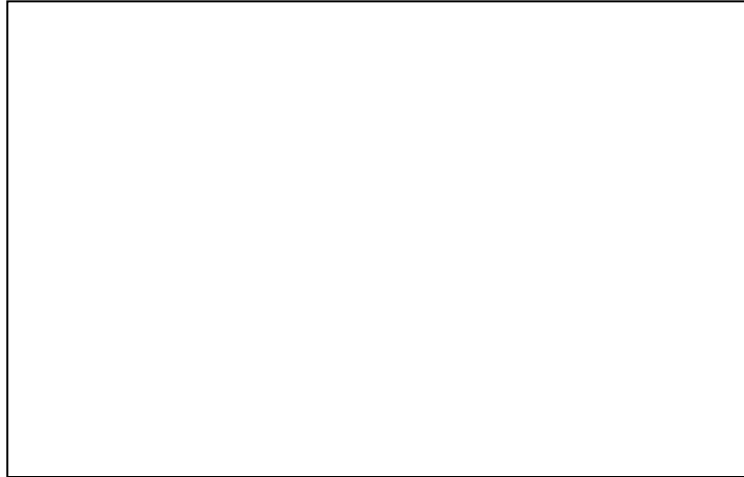

Nom i Cognoms:

Tots els exercicis tenen el mateix pes.

Exercici 1

Aquí teniu una llista d'etapes/tasques del pipeline gràfic, ordenades per ordre alfabètic. Torna-les a escriure a la dreta, però ordenades segons l'ordre al pipeline gràfic:

- Fragment Shader (FS)
- Geometry Shader (GS)
- Rasterització
- Stencil test
- Vertex Shader (VS)

**Exercici 2**

Les variables *varying* són de sortida en un VS i d'entrada en un FS. En quina de les següents etapes **s'interpolen** els valors de les variables *varying*?

- (a) Fragment Shader (FS)
- (b) Rasterization
- (c) Primitive assembly
- (d) Viewport transformation

Exercici 3

Volem aplicar un escalat uniforme 2x als vèrtexs d'un objecte. L'aplicació concreta fa servir coordenades homogènies, i ens diuen que no podem assumir que la coordenada *w* serà sempre 1. Indica, per cada opció, si és una forma correcta o no d'aplicar l'escalat anterior a `gl_Vertex`:

- (a) `vec4 v = gl_Vertex * 2.0;`
- (b) `vec4 v = vec4(gl_Vertex.xyz * 2.0, gl_Vertex.w);`
- (c) `vec4 v = gl_Vertex.xyz * 2.0;`
- (d) `vec4 v = gl_Vertex * vec4(2.0, 2.0, 2.0, 1.0);`

Exercici 4

Aquest fragment de codi calcula el vector V que es necessita pels càlculs d'il·luminació:

```
vec3 V = normalize( (gl_ModelViewMatrix * (Obs - Vert)).xyz );
```

(a) En quin espai estan Obs (posició de l'observador) i Vert (posició del vèrtex)?

(b) En quin espai està V?

Exercici 5

Completa el següent FS per tal que calculi correctament la contribució difosa:

```
varying vec3 normal; // eye space
```

```
varying vec3 pos; // pos en eye space
```

```
vec4 lightSource( vec3 N, vec3 V, gl_LightSourceParameters light ) {
```

```
    vec3 L = normalize( light.position.xyz - V );
```

```
    vec3 R = normalize( 2.0*dot(N,L)*N-L );
```

```
    V = normalize (-V.xyz);
```

```
    float diff; // cal que el calculeu vosaltres
```

```
    diff =
```

```
    float spec; float RdotV = max( 0.0, dot( R,V ) );
```

```
    spec = pow( RdotV, gl_FrontMaterial.shininess );
```

```
    return gl_FrontMaterial.diffuse * light.diffuse * diff +
```

```
        gl_FrontMaterial.specular * light.specular * spec;
```

```
}
```

Exercici 6

Es poden modificar les coordenades (x,y) d'un fragment (gl_FragCoord.xy) en un FS?

Exercici 7

Si un punt és interior al frustum de visió, les seves coordenades en NDC estaran

(a) Entre 0 i 1, amb z=0 pels punts sobre el pla de retallat anterior

(b) Entre 0 i 1, amb z=1 pels punts sobre el pla de retallat anterior

(c) Entre -1 i 1, amb z=-1 pels punts sobre el pla de retallat anterior

(d) Entre -1 i 1, amb z=1 pels punts sobre el pla de retallat anterior

Exercici 8

Tenim una textura de 256x256. Quina és l'amplada d'un texel en espai normalitzat de textura?

- (a) $\Delta s = 256$
- (b) $\Delta s = 1/256$
- (c) $\Delta s = 1/(2*256)$
- (d) $\Delta s = \log(256)$

Exercici 9

Respecte a la diferència entre calcular la il·luminació per vèrtex (al VS) o fer-ho per fragment (al FS), contesta CERT/FALS a cada pregunta:

- (a) Fer-ho per fragment té generalment més qualitat (la simulació és més acurada)
- (b) Fer-ho per fragment incrementa el número de variables varying a interpolar

Exercici 10

Si fem servir un GS per dividir cada triangle en quatre triangles, generant nous vèrtexs, on té sentit que es facin els càlculs d'il·luminació?

- (a) Al VS, GS o FS, indistintament
- (b) Al VS o al FS, indistintament
- (c) Al GS o al FS, indistintament
- (d) Només es podrà fer al FS.

Exercici 11

Indica, per cadascuna d'aquestes condicions relacionades amb l'estat d'OpenGL, si pot afectar negativament al funcionament correcte de la selecció d'objectes basada en lectura del buffer de color (pseudocolor). Contesta SI (podria deixar de funcionar) o NO (no afecta):

- (a) `glDisable(GL_DEPTH_TEST)`
- (b) `glEnable(GL_LIGHTING)`
- (c) `glEnable(GL_BLEND);`
- (d) `glClearColor(0,0,0,0);`

Exercici 12

Tenim una aplicació que només treballa amb escenes de fins a 24 primitives. Volem poder seleccionar primitives amb el mouse, però volem seleccionar totes les primitives sota el mouse (no només la visible). Una forma de fer-ho és utilitzar la tècnica basada en lectura del buffer de color. La idea seria codificar el color de cada primitiva fent que cada bit del color RGB correspongui a una primitiva, de forma que el color RGB tindrà, en binari, tot 0's excepte un bit a 1 que correspon a la primitiva (en decimal, els colors RGB seran potències de 2). Quina funció de blending caldria fer servir, per tal de representar al buffer de color tots els objectes que s'ha projectat a cada píxel?

- (a) `glBlendFunc(GL_ONE, GL_ONE);`
- (b) `glBlendFunc(GL_ZERO, GL_ONE);`
- (c) `glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA)`
- (d) `glBlendFunc(GL_SRC_ALPHA, GL_ZERO);`

Exercici 13

Volem dibuixar les ombres que projecten els objectes sobre el pla $Z = 5$, amb la tècnica d'ombres per projecció. Suposant una llum infinitament allunyada en direcció de l'eix Z, indica per quina matriu 4x4 cal multiplicar la posició del vertex en world space per tal d'obtenir les ombres projectades al dibuixar els objectes.

Exercici 14

Disposem d'una funció `sampleSphereMap` que, donat un vector unitari R, i una textura que representa un sphere map, ens retorna el color de l'entorn en la direcció R. Completa el següent shader per tal que implementi sphere mapping, amb els càlculs en object space:

```
uniform sampler2D spheremap;
uniform vec3 obs; // posició de l'observador en object space
varying vec3 P; // posició del fragment en object space
varying vec3 N; // normal en object space

vec4 sampleSphereMap(sampler2D sampler, vec3 R); // funció que podeu cridar; R en object space

void main() {
    vec3 R;

    [Empty box for code completion]

    gl_FragColor = sampleSphereMap(spheremap, R); }
```

Exercici 15

Aquest fragment de codi per generar reflexions és incomplet. Quina crida OpenGL (relacionada amb les matriu de transformació geomètrica) manca i on caldria afegir-la?

// 1. Dibuixem el mirall a l'stencil buffer

```
glEnable(GL_STENCIL_TEST);  
glStencilFunc(GL_ALWAYS, 1, 1);  
glStencilOp(GL_KEEP, GL_KEEP, GL_REPLACE);  
glDepthMask(GL_FALSE); glColorMask(GL_FALSE...);  
dibuixar(mirall);
```

// 2. Dibuixem els objectes en posició virtual

```
glDepthMask(GL_TRUE); glColorMask(GL_TRUE...);  
glStencilFunc(GL_EQUAL, 1, 1);  
glStencilOp(GL_KEEP, GL_KEEP, GL_KEEP);  
glPushMatrix();  
glLightfv(GL_LIGHT0, GL_POSITION, pos);  
glCullFace(GL_FRONT);  
dibuixar(escena);  
glPopMatrix();
```

// 3. Dibuixem el mirall semitransparent

```
glDisable(GL_STENCIL_TEST);  
glLightfv(GL_LIGHT0, GL_POSITION, pos);  
glCullFace(GL_BACK);  
dibuixar(mirall);
```

// 4. Dibuixem els objectes reals

```
dibuixar(escena);
```

Exercici 16

Contesta les següents preguntes en relació al rendiment de diferents tècniques de simulació d'ombres:

(a) Sombres del projecció amb stencil, una font de llum estàtica: quantes vegades cal pintar l'objecte ocluser cada frame?

(b) Shadow mapping, dues fonts de llum dinàmiques, amb un FS: quantes vegades cal pintar l'escena cada frame?

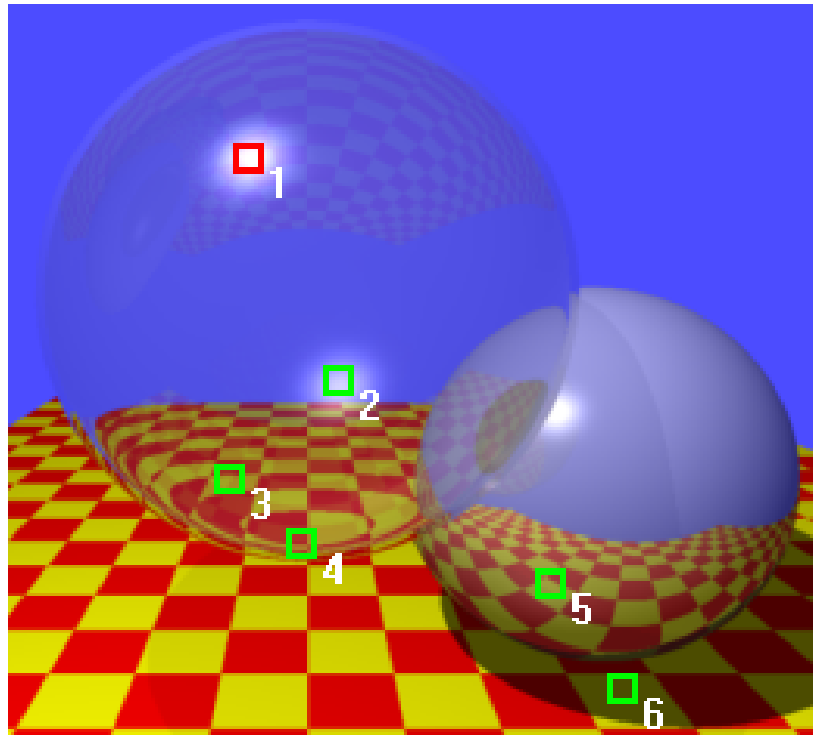
Exercici 17

Indica quins d'aquests camins seria capaç de simular una implementació de l'algorisme clàssic de raytracing amb nivell màxim de recursivitat 10:

- a) LSDE
- b) LSSSDE
- c) LSSSSSSSDE
- d) LDSSE

Exercici 18

Aquesta figura s'ha generat amb raytracing clàssic:



Indica quins píxels (1-6) podrien veure afectat significativament el seu color final si canviem l'índex de refracció de l'esfera semi-transparent:

Preguntes per a l'avaluació de les competències transversals

Pregunta 1

A què fa referència el terme *haptic*?

- (a) Interacció en aplicacions mèdiques
- (b) Interacció explícita
- (c) Manipulació d'objectes deformables
- (d) Realimentació tàctil

Pregunta 2

Indica al menys un parell de tècniques d'adquisició de dades de volum en medicina.