Examen Parcial de Gràfics Curs 2013-14 Q1

Nom i Cognoms:

Exercici 1

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

```
vec3 L = normalize( gl_LightSource[0].position.xyz - Pos);
```

En quin espai(s) pot estar *Pos* per què el càlcul sigui correcte?

Exercici 2

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:

- Depth test
- Il·luminació per fragment
- Rasterització
- Transformació a clip space

Exercici 3

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 4

Indica quins dos valors cal que el VS passi al FS per tal d'aplicar el model de Phong al FS.

Exercici 5

Aquesta línia sovint apareix al codi que avalua la component difosa de la il·luminació:

```
float NdotL = max( 0.0, dot( N,L ) );
```

Explica per què cal fer el màxim.

Exercici 6

Què és incorrecte a la última línia d'aquest codi GLSL?

```
uniform sampler3D sampler;
...
gl_FragColor = texture3D(sampler, gl_TexCoord[0].st);
```

Exercici 7

Aquest fragment de codi OpenGL està fent servir la generació automàtica de coordenades de textura amb el mode GL_OBJECT_LINEAR:

```
GLint s_plane[4] = { 2, 0, 0, 0 };
GLint t_plane[4] = { 0, 0, 2, 0 };
glTexGeni(GL_S, GL_TEXTURE_GEN_MODE, GL_OBJECT_LINEAR);
glTexGeniv(GL_S, GL_OBJECT_PLANE, s_plane);
glTexGeni(GL_T, GL_TEXTURE_GEN_MODE, GL_OBJECT_LINEAR);
glTexGeniv(GL_T, GL_OBJECT_PLANE, t_plane);
glEnable(GL_TEXTURE_GEN_S);
glEnable(GL_TEXTURE_GEN_T);
```

Quines coordenades de textura (s,t) es calcularan pel vèrtex (10, 0, 5)?

Exercici 8

Tenim una textura de 512x512. Quina és l'amplada d'un texel en espai normalitzat de textura?

(a) $\Delta s = 512$

(b) $\Delta s = 1/512$

(c) $\Delta s = 1/(2*512)$

(d) $\Delta s = \log(512)$

Exercici 9

Quan la projecció d'una primitiva texturada resulta en minification, què podem dir de la preimatge associada a un fragment?

Exercici 10

Quants texels d'una textura 2D s'han de consultar per prendre una mostra de la textura amb filtrat bilinial?

Exercici 11

La matriu necessària per implementar projective texture mapping és pot obtenir amb aquest codi OpenGL:

```
glLoadIdentity();
glTranslated(0.5, 0.5,0.5);
glScaled(0.5, 0.5, 0.5);
gluPerspective(…);
gluLookAt(…);
```

Quins paràmetres cal usar per les crides gluPerspective i gluLookAt?

(a) Els paràmetres del projector per gluPerspective, i els paràmetres de la càmera per gluLookAt

(b) Els paràmetres de la càmera, en tots dos casos

(c) Els paràmetres del projector, en tots dos casos

(d) Els paràmetres de la càmera per gluPerspective, i els paràmetres del projector per gluLookAt

Exercici 12

Si (T, B, N) són els vectors tangent, bitangent i normal, expressats en object space, indica quina és la matriu que ens permet passar de tangent space a eye space, considerant aquesta definició:

```
mat4 M = mat4(T, B, N, vec4(0,0,0,1));
```

(a) glModelViewMatrix * M

(b) M

(c) M * glModelViewMatrix

(d) glNormalMatrix * M

Exercici 13

Si implementem la tècnica de bump mapping fent servir un normal map, les normals codificades al normal map estaran en

(a) object space

(b) world space

(c) clip space

(d) tangent space

Exercici 14

Si implementem bump mapping, com cal declarar les variables que representen els vectors tangent i bitangent, en GLSL?

(a) attribute pel VS, varying pel FS

(b) uniform pel VS, varying pel FS

(c) uniform pel VS, attribute pel FS

(d) attribute pel VS, no ni cal al FS

Exercici 15

Escriviu codi en llenguatge GLSL per passar un punt P de *eye space* a *model space*

```
vec4 P;
...
P = gl_ModelViewMatrixInverse * P;
```