 Nome ____ Número ____________

COMPUTAÇÃO GRÁFICA E INTERFACES

Ano letivo 2016/2017 – Test 1 – 2016.11.03

Notice

The backsides are considered as draft area. If you place an answer there, don't forget to mention it on the front side.

Do not remove the staple!
Duration: **1H30**!

1. (3/20)

Mark the following sentences with T(True) or F(False). Each wrong answer will deduct 50% of its score. In the WebGL visualization pipeline:

In the end, the fragment shader may not even be executed at all when we ask to draw a triangle, for instance.	
The javascript application sends pixels to the pipeline and the GPU is responsible for painting them through the execution of a fragment shader.	
We can associate values (colours, normal vectors, etc.) to the individual primitives that are produced by the application.	
A fragment shader can receive arbitrary data that is not the output produced by a vertex shader.	
An application always uses the data type **vec4** to assign coordinates to the vertices.	
We can change a GLSL program (vertex + fragment shader) in the middle of a primitive.	
Uniform variables correspond to values that the application sends to the vertex shader exclusively.	
varying variables contain values known by the javascript application.	
The output of a vertex shader is completely defined by the user without any imposition of producing any specific type of output.	
The output of a fragment shader always needs to contain the colour of the pixel, unless the fragment is discarded.	

2. (3/20)

The following shaders are part of the same GLSL program.
a) complete the code by filling the blanks:

```
uniform ____ m;
_________ ____ vPos;
_________ ____ vfoo;
varying vec4 ffoo;
void main() {
    ___________ = m * vPos;
    ____ = vec4(vfoo, 1.0);
}
```

```
_______ ____ ffoo;

void main() {
    ____________ = ffoo;
}
```

b) In your opinion, what could be the meaning/usage of the variable `vfoo`:

 Nome ______________________________ Número __________

3. (4/20)

A screen of a tablet has a resolution of 1280x800 pixels. The left side is dedicated to text presentation. The right side is further divided into two equally sized areas, one on top of the other. A 2D window, defined in World Coordinates (WC) by its limits $x_{min} \leq x \leq x_{max}$ e $y_{min} \leq x \leq y_{max}$, is to be mapped, simultaneously to the top right and bottom right quadrants of the display, without clipping or deforming its contents. The viewports are to be centred in their respective available areas.

a) What are the dimensions of the viewports, knowing that the window has an aspect ratio of 4:3? You don't need to perform any calculations and you can simply present the respective expressions.

b) What is the window to viewport transformation **M** of the top right viewport, presented in the form of a natural composition of elementary 2D geometric transformations (T, S or R), to be used in the form: **p'=M.p**:

M =

c) What would be the necessary geometric transformation to offer the user a picking operation (performed in object space) in the viewport mentioned in b)?

M_{pick} =

d) Imagine that the window aspect ratio is now 16:10. What would be new window to viewport transformations for each of the viewports? In your answer provide compositions as identical as possible.

$M_{topright}$ =

$M_{bottomright}$ =

4. (3/20)

Consider the elementary geometric transformations studied, to be used in the form: **p'=M.p**.

a) for each of the following sequences of 3D transformations, **provide only one equivalent compositions**, prioritized in the following manner: (**1st**) a simpler composition; (**2nd**) alternative composition; (**3rd**) the initial composition (otherwise).

1. S(1,3,4).S(2,5,1).T(1,0,0) ______________________________
2. R_x(20°).R_x(-30°).T(2,2,2).R_x(50°) ______________________________
3. S(2,3,3)R_x(20°)R_y(30)S(1,3,1) ______________________________
4. T(2,5,1).R_x(270°).S(1,2,2).R_x(90°) ______________________________

b) consider a line in 2D defined by the equation $y=mx+b$, where m the slope and (0,b) a point on the line. We wish to offer a `mirror(m,b)` operation to the programmer that will reflect objects using the given line as the axis of symmetry. Provide a composition of elementary geometric transformations in 2D that will achieve the intended effect. Please instantiate all the parameters of your solution.

M_{mirror} =

5. (4/20)

a) Consider a point P in 3D, with homogeneous coordinates (2,-2,4,1). Fill the table below with the **3D coordinates of the images** (and not the 2D coordinates on the projection plane) of the points after each of the stated projections:

Front view	Top view	Left view	Right view

b) Consider the following additional points, also in homogeneous coordinates, Q=(2,-2,0,1), R=(3,-2,0,1) and S=(6,0,0,3). Fill the table below with the length and orientation (angle formed with the horizontal) for each of the presented line segments, after an oblique projection with the following parameters: l=0.4, α=30°:

Line Segment	**Length**	**Orientation**
PQ		
QR		
QS		
RS		Don't fill this space.

c) if, in the projection referred in b), we used *l*=1, what could you say about the angle formed between the projection lines and the projection plane? Would it be bigger/smaller/unchanged? Justify your answer!

6. (3/20)

Consider the methods studied for hidden surface removal:

a) State both an advantage and a disadvantage of using the back face culling method instead of the z-buffer. Justify your opinion!

b) Admitting that the scene to be visualized is composed of several thousand small polygons, is there any specific rendering order that would speed up the rendering process?