Please read these instructions carefully!

- Answer the test questions in the separate answer sheet.
- You may use the back of all the paper sheets as drafting area (*rascunho*).
- How to:

	A	B	C	D	E
Select the answer (A):	(●	○	○	○	○)
Replace the answer (A) by (C):	(✗	○	●	○	○)
Cancel (C) and reactivate (A):	(◉	○	✗	○	○)

- This test has 25 QUESTIONS (ignoring question zero), each question has a score of 200/25 points.
- **POINTS LOST** for each wrong answer (in percentage of the question's points):
  $\sum wrong$ $1 = 0\%$ $2 = 11.11\%$ $3 = 22.22\%$ $\geq 4 = 33.33\%$.

Name: ______________________________ Number: ______________

0. Test version. This is not a question and will be ignored for grading!

   A. ⇒ ● ⇐ B. C. D.

Consider the pipeline of a modern graphics card, used by a 3D WebGL program.

1. Choose the most appropriate description for the task that gets executed in the last programmable stage of the pipeline by the respective shader.

   A. Decide if a pixel is to be painted or not and, in the first case, assign it a color.

   B. Always assign a color to the pixel since the decision of whether the pixel is going to be painted or not is only performed later during the z-buffer test.

   C. Decide if a pixel is to be painted or not and, in both cases, assign it a color.

   D. Determine if a pixel is to be painted or not and, in the first case, determine its location on the screen.

2. Choose the correct stage of the pipeline where the visibility test is performed.

   A. Right after the rasterisation stage.

   B. After the execution of the fragment shader.

   C. Immediately after the vertex shader is applied.

   D. Immediately before the fragment shader is applied.

3. Choose the information that best describes what is associated with a vertex at the exit of the vertex processor.

   A. A vertex location in Clip Coordinates and a set of 0 or more outputs to be interpolated during the rasterisation stage.

   B. A vertex location in Clip Coordinates and a depth value that is stored in the Z-Buffer.

   C. A vertex location in Clip Coordinates.

   D. A vertex location in Clip Coordinates and a colour value that is interpolated later during rasterisation.

4. Choose the correct sentence that applies to a WebGL application by the time it asks to render some set of primitives through a call to `drawArrays()` or `drawElements()`.

   A. Vertex data is transfered from central memory to GPU memory as a consequence of the call to `drawArrays()` or `drawElements()`.

   B. Vertex data needs to be present in GPU memory.

   C. Vertex data needs to be present in CPU (central) memory.

   D. Vertex data needs to be present both in CPU and GPU memory.

In the following questions we will consider GLSL (Graphics Library Shading Language) - The WebGL shading language. A GLSL program will be a pair of shaders that are used simultaneously in the pipeline while an application will be the whole program, including its WebGL javascript part.

5. Choose the option that describes the correct semantics for the value of a variable declared in a vertex shader with the `attribute` qualifier.

   A. None of the other alternatives is correct.

   B. Its value always varies from fragment to fragment, by linear interpolation of the computed values for each vertex of the primitive being rendered.

   C. Its value is assigned a single common value to all vertices in the javascript side and it remains fixed during the GLSL program execution.

D. **Its value may be different, from vertex to vertex, and it is extracted from a buffer filled by the javascript side.**

6. Choose the correct option that applies to a `uniform` variable.

   A. Its value may change while a call to `drawArrays()` or `drawElements()` is being handled.

   B. **It can be declared in both shaders of a GLSL program.**

   C. Its value may never change during the execution of the application.

   D. It can only be declared in one the shaders of a GLSL program.

7. Consider the GLSL (Graphics Library Shading Language) piece of code presented on the right. Local variables '`c`' and '`p`' have previously been declared with the correct type to make the presented code valid. What is the value that gets assigned to '`p`'?

```
c = 4.0 * vec3(0.25, 2.0, 1.0);
p = c.rbg * vec3(4.0, c.b, 6.0);
```

   A. **(4.0, 16.0, 48.0)** B. (4.0, 24.0, 48.0) C. (4.0, 32.0, 24.0) D. (4.0, 48.0, 16.0)

8. Consider the GLSL program presented on the right. It is used by an application that draws a model of a car engine where each pixel will be animating its color from an initial value to a final one. The colors are assigned by the javascript application on a per vertex basis. Which of the following sequences can be used to fill the blanks in the code so that the application works as described?

   A. `float, attribute, uniform, vec4, c, c`

   B. `float, varying, uniform, vec4, x, x`

   C. `vec3, varying, varying, vec3, x, x`

   D. `vec3, attribute, varying, vec3, c, c`

```
// vertex shader
attribute vec4 v;
attribute ____ y;
_________ vec3 x;
uniform float time; //in [0,1] range
_________ ____ c;

main()
{
  gl_Position = v;
  c = mix(x, y, time);
}

// fragment shader
varying vec3 ____;
main()
{
  gl_FragColor = vec4( ____, 1.0);
}
```

9. A WebGL application will be used to simulate a single rocket in 2D that is launched at point $p = (x_0, y_0)$ at instant t_0, with initial velocity v_0 and explodes in the air at instant t_1. Each piece k will afterwards continue with a new initial velocity given by $v_1^{(k)}$. The simulation is performed by displaying a fixed number of points (thousands) with the `gl.POINTS` primitive and using only one GLSL program. Each point will represent a small piece of the rocket and its trajectory is governed (before and after the explosion) by the set of equations on the right. Note that even before the explosion, we can display the rocket by using its thousands of smaller pieces as long as they move together. Choose the option that best describes how you would send the data from your application to the GLSL program.

$$x = x_i + v_{i_x} T$$
$$y = y_i + v_{i_y} T - \frac{g}{2} T^2$$

   A. **During initialization stage fill a buffer with the v_1 attribute for each piece and set uniform constants t_0, t_1, v_0 and p. Bef**

   B. During the initialization stage fill a buffer with the p, v_0 and v_1 attributes for each piece and set uniform constants t_0 and t_1. Before each rendering step set uniform constant T.

   C. During the initialization stage fill a buffer with th v_0 attribute for each piece and set uniform constants t_0, t_1 and p. Before each rendering step set uniform constant T. The shaders will determine $v1$ for each piece.

   D. None of the other options will make the application work as intended.

Consider the 3D points and vectors in the table on the right represented using 4D homogeneous coordinates.

A	(2, 1, 6, 2)
B	(0, 3, 4, 0)
C	(2, 2, 4, 1)
D	(0, 3, 0, 6)
E	(1, 2, 1, 0)

10. Choose the **false** statement below.

    A. **B can have multiple alternative representations.**

    B. A can have multiple alternative representations.

    C. B and E are vectors.

    D. A, C, and D are points.

11. What is the result of the expression: C + B - E ?

    A. (1, 3, 7, 0) **B.** **(1, 3, 7, 1)** C. The vector (1, 3, 7) D. The operation is not valid

12. What is the result of the expression: D - A + E?

    A. **Vector (0,2,-2)** B. (-1, 4, -5, 4) C. Point (0,2,-2) D. Point (-1/4, 1, -5/4)

13. What is the advantage of using homogeneous coordinates in the graphics card hardware?

   A. There is no advantage and it is just a matter of design.

   B. All geometrical transformations are handled in the same way.

   C. The operations become more efficient.

   D. It saves space in memory.

---

14. Choose the option below where the geometrical operations could have been swapped without affecting the result.

   A. $S(3, 3, 1).R_z(\alpha)$

   B. $T(4, 0, 1).R_y(\alpha)$

   C. $T(-4, 5, 2).S(1, 2, 2)$

   D. $R_x(\alpha).S(1, 2, 1)$

15. An instance of a primitive object has been transformed by the composite transformation $S(1, 4, 4) \cdot R_x(-30) \cdot T(2, 4, 2) \cdot R_y(15)$. Choose the composite transformation that would undo the original transformation.

   A. $S(-1, -4, -4) \cdot R_x(30) \cdot T(-2, -4, -2) \cdot R_y(-15)$

   B. $R_y(15) \cdot T(2, 4, 2) \cdot R_x(-30) \cdot S(1, 4, 4)$

   C. $S(1, 1/4, 1/4) \cdot R_x(30) \cdot T(-2, -4, -2) \cdot R_y(-15)$

   D. $R_y(-15) \cdot T(-2, -4, -2) \cdot S(1, 1/4, 1/4) \cdot R_x(30)$

16. Choose the sequence below that is normally known as an instatiation transformation, used in 2D modelling software that restricts it to 3 elementary transformations.

   A. $T(..., ...)R(...)S(..., ...)$

   B. $S(..., ...)R(...)T(..., ...)$

   C. $S(..., ...)R(...)S(..., ...)$

   D. $T(..., ...)S(..., ...)T(..., ...)$

A computer screen with UHD-1 resolution (3840 × 2160 pixels [16:9]) is going to be used to display the contents of a window defined in 2D world coordinates by its limits $-200 \leq x \leq 10$ and $400 \leq y \leq 490$. The viewport will be centered on the screen and it should maximize the viewing area without clipping the window contents. As usual, the origin of the 2D coordinate system associated with the device has its origin located in the top left corner of the screen.

17. What are the viewport dimensions in pixels?

   A. $3840 \times (\frac{210}{90 \times 3840})$ B. $\frac{90}{210 \times 2160} \times 2160$ **C.** $3840 \times (\frac{90 \times 3840}{210})$ D. $\frac{210 \times 2160}{90} \times 2160$

18. What would you choose as the first operation to be performed by the window to viewport transformation?

   A. $T(-1920, -1080)$ **B.** $T(95, -445)$ C. $T(-3840, 0)$ D. $T(-95, -445)$

19. What would you choose as the last operation to be performed by the window to viewport transformation?

   A. $T(3840, 0)$ **B.** $T(1920, 1080)$ C. $T(-95, 445)$ D. $T(95, 445)$

20. What would you change in the problem specification to visualize the same content in the same location of the screen but using its full 3840 × 2160 viewing area?

   A. The scaling factor included in the window to viewport transformation.

   B. The final translation in the window to viewport transformation.

   C. The upper and lower limits of the window.

   D. The left and right limits of the window.

21. Consider the Z-buffer algorithm for Hidden Surface Removal. Choose the sentence that is **false**.

   A. The resolution of the Z-buffer needs to be at least the resolution of the used screen space.

   B. The Z-buffer needs to be cleared at the start of each frame even when double buffering is being used.

   C. The order by which the objects are processed is irrelevant in terms of speed.

   D. The order by which the objects are processed is irrelevant for the final image.

22. The back face culling algorithm is enough to solve the problem of HLHSR when applied to certain scenes. Choose the option where this algorithm alone would **solve** the HLHSR problem.

A. Several non convex solids that don't overlap in the projection.

B. **A cube and a cone that overlap in the projection but not in 3D space and that are drawn in back to front order.**

C. A cube and a cone that overlap in the projection but not in 3D space and that are drawn in front to back order.

D. A single cylinder solid with faces both positively and negatively oriented.

Consider the following points in 3D: $P(2, -2, 4)$, $Q(2, -2, 0)$, $R(3, -2, 0)$ and $S(2, 0, 0)$ and the line segments $\overline{PQ}$, $\overline{QR}$ and $\overline{QS}$.

23. What is the location on the projection plane (x_p, y_p) of P' when projected using a top view?

A. $(-2, -4)$ B. $(-2, 4)$ **C.** $(2, 2)$ D. $(2, -2)$

Esta questão foi considerada corretecta em todos os alunos.

24. Which segment would appear larger under an oblique projection with $l = 0.5$ and $\alpha = 30$?

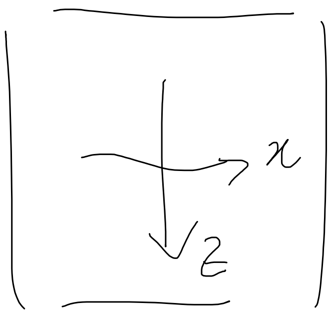
A. **$\overline{P'Q'}$ and $\overline{Q'S'}$.** B. $\overline{P'Q'}$. C. $\overline{Q'S'}$ and $\overline{Q'R'}$. D. $\overline{Q'S'}$.

25. What is the angle $P'\hat{Q}'R'$ after an oblique projection with the parameters $l = 0.5$ and $\alpha = 30$:

A. 150° B. 30° C. 120° D. 90°

//

23. Planta



ou seja $x_p = x$

$y_p = -z$

logo a resposta seria $(2, -4)$, a qual não aparece nas opções apresentadas.

//