Assignment 10: compile the shaders code

In this Assignment you have to write a Vertex and a Fragment Shader that create a procedural surface with algorithmic defined colors. In particular, in the vertex shader, you have to add code in the highlighted session:

```
#version 450
#extension GL_ARB_separate_shader_objects : enable

layout(set = 0, binding = 0) uniform UniformBufferObject {
        mat4 mvpMat;
        mat4 mMat;
        mat4 nMat;
} ubo;

layout(set = 0, binding = 1) uniform GlobalUniformBufferObject {
        float time;
} gubo;

layout(location = 0) in vec3 inPosition;
layout(location = 1) in vec3 inNorm;
layout(location = 2) in vec2 inUV;

layout(location = 0) out vec3 fragPos;
layout(location = 1) out vec3 fragNorm;
layout(location = 2) out vec2 fragUV;

void main() {
        float x = inPosition.x;
        float y = 0.0f;
        float z = inPosition.z;
        float t = gubo.time;

        /***** Insert the code here to compute the correct value of y ****/

        /***** Leave it as is from this point on ****/

        vec3 vpos = vec3(x, y, z);

        gl_Position = ubo.mvpMat * vec4(vpos, 1.0);
        fragPos = (ubo.mMat * vec4(vpos, 1.0)).xyz;
        fragNorm = inNorm;
        fragUV = inUV;
}
```

that computes the y coordinate of the surface with the following algorithm:

$$c_x = \frac{\sin\frac{t}{3}}{2}$$

$$c_z = \frac{\cos\frac{t}{3}}{2}$$

$$l = \sqrt{(x - c_x)^2 + (z - c_z)^2}$$

$$y = y + \sin\left(\frac{t}{4} + 4 \cdot l\right) \cdot e^{-\frac{l^2}{4}}$$

Similarly, you should add the code in this highlighted session of the fragment shader:

```
#version 450
#extension GL_ARB_separate_shader_objects : enable

layout(location = 0) in vec3 fragPos;
layout(location = 1) in vec3 fragNorm;
layout(location = 2) in vec2 fragUV;

layout(location = 0) out vec4 outColor;
layout(set = 0, binding = 1) uniform GlobalUniformBufferObject {
        float time;
} gubo;

void main() {
        vec3 Norm = fragNorm;

        float y = fragPos.y;
        float r = 0.0f, g = 0.0f, b = 0.0f;
        float t = gubo.time;

        /***** Insert the code here to compute the correct value of r, g and b ****/

        /***** Leave it as is from this point on ****/

        outColor = vec4(r, g, b, 1.0);
}
```

that computes the r, g and b output colors with the following algorithm:

$$h = \mathrm{mod}\left(\frac{t}{3} + y, 6\right)$$

$$\begin{cases} r = 1 \\ g = h \quad \mathit{if}\ h < 1 \\ b = 0 \end{cases}$$

$$\begin{cases} r = 2 - h \\ g = 1 \quad \mathit{if}\ 1 \leq h < 2 \\ b = 0 \end{cases}$$

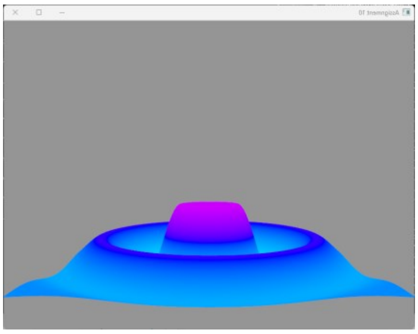
$$\begin{cases} r = 0 \\ g = 1 \quad \mathit{if}\ 2 \leq h < 3 \\ b = h - 2 \end{cases}$$

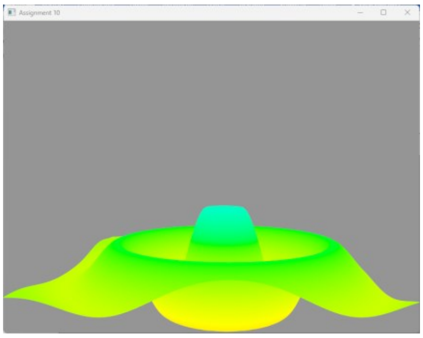
$$\begin{cases} r = 0 \\ g = 4 - h \quad \mathit{if}\ 3 \leq h < 4 \\ b = 1 \end{cases}$$

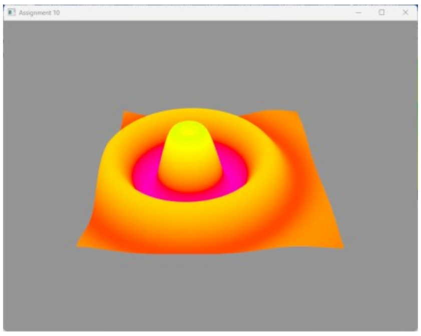
$$\begin{cases} r = h - 4 \\ g = 0 \quad \mathit{if}\ 1 \leq h < 2 \\ b = 1 \end{cases}$$

$$\begin{cases} r = 1 \\ g = 0 \quad \mathit{if}\ h \geq 5 \\ b = 6 - h \end{cases}$$

The expected result should generate images similar to the following:




Assignment 10


Assignment 10