Advanced Graphics - Paractical sheet 1

Go to Shadertoy.com. Type ‘supervisionadvgfx’ in the search box and open shader named ‘Adv GFX Cam- Supervision AB’. You should see the following:

Familiarize yourself with the code.
In this set of exercises we will focus on signed distance fields, illumination models, Vornoi diagrams, separable kernels, tone mapping and inverse display model.

Ex 1.
Go to “Buf A” tab, scroll down to PhongModel method. Change color of the sphere to green. Implement Phong illumination model inside PhongModel(). The final result should look similar to this:
Ex 2.
Add necessary code to TorusSDF(). Replace the sphere in the middle of the scene with the torus (make sure that the torus is aligned with xy plane).

Ex 2a.
Make torus rotate about x and z axes. You can access the time using "iTime" variable provided by the system.

Ex 2b.
Populate CylinderSDF() and draw the cylinder next to the torus. Use min function.
Ex 2c.
Combine box and sphere SDF to achieve similar results to this:

Ex 2d.
Implement twist operation and apply it the cut box shape. Your results should look similar to this:

Ex 3.
Add Blinn-Phong Shading model. Compare the results with previously implemented Phong. Save your shaders - we will go back to them.
Ex 4.
Create new shader in shadertoy. Name it Vornoi_[your CRSID]
Populate your shader with the following code:

```glsl
const vec2 Points[9] = vec2[]
(
    vec2(-1.3,0.6), vec2(-0.2,0.3), vec2(0.1,0.8),
    vec2(0.6,0.5), vec2(1.3,-0.2), vec2(0.2,0.4),
    vec2(0.4,-0.9), vec2(-0.3,-0.9), vec2(-0.5,-0.61)
);

const vec3 Colors[9] = vec3[]
(
    vec3(0.1,0.2,1.0), vec3(1.0,0.3,0.5), vec3(0.0,1.0,0.0),
    vec3(1.0,0.2,0.0), vec3(1.0,1.0,0.5), vec3(1.0,1.0,0.0),
    vec3(0.1,0.2,0.0), vec3(0.98,0.5,0.0), vec3(0.0,0.8,0.3)
);

void mainImage( out vec4 fragColor, in vec2 fragCoord )
{
    /// Normalized pixel coordinates (from 0 to 1)
    vec2 uv = fragCoord.xy/iResolution.xy;

    // Move coords to -1..1 space
    uv = uv * 2.0 - 1.0;

    // aspect ratio
    uv.x *= iResolution.x/iResolution.y;

    vec3 col = vec3(0.0);
    fragColor = vec4(col,1.0);
}
```

Your task is to implement Vornoi diagram. Final result will look similar to this:

Try animating the diagram to achieve results comparable to those shown here.
Save your results.

Ex 5.
Go back to your ray marched scene and create another buffer. Copy your Vornoi code there and render it to texture. In “Buf A” pass apply this texture to the floor. The results that we are looking for look is this:

Ex 6.
Implement low-pass (spatial frequency) box filter with separable kernel of [0.2,0.2,0.2,0.2,0.2,0.2]. You will need to use two rendering passes. Horizontal pass has been partially implemented for you (Buf B). Use buff C to implement vertical pass.

Ex 6a.
Change size of the kernel to 9. Do you see any changes?

Ex 6b.
Using gaussian kernel calculator, implement 9x9 gaussian filter with sigma value of 11.

Ex 7.
Replace gamma correction with more sophisticated Gamma offset gain model (GOG).