Instancing

1. Why did you implement instancing?

   A. The reason why I implement instancing is that it seems a basic optimization of graphic. Actually, I have had a doubt that how my computer cannot be slow while a chunk of rock and a tons of leaf are drawn. The answer of my doubt was instancing. Furthermore, I believed it can be used and adopted easily on my gam200 project. I choose instancing as a 'A grade'

2. What is instancing?

   A. Briefly, it is a drawing technique that a chunk of object via only one call of draw function. If I give GPU an array of data, it draw same object that has same texture data with different a chunk of data kind of translation, size, and color. Fortunately, OpenGL already has a support function for instancing. What I have to do is to understand what instancing is and add a few of send uniform functions.

3. How did you implement it?

```
void Sketch::InstancingInit()
{
    Graphics::Color4f color{ grey: 0.f };
    int index = 0;
    for (int y = -450; y < 90000; y+=120)
    {
        for (int x = -750; x < 850; x += 150)
        {
            if (index >= maxSizeInstancing)
            {
                return;
            }
            translations[index] = vector2<float>(x: static_cast<float>(x), y: static_cast<float>(y));
            colors[index] = vector3<float>{
                x: static_cast<float>(rand() % 255)/255.f,
                y: static_cast<float>(rand() % 255)/255.f,
                z: static_cast<float>(rand() % 255)/255.f };
            scales[index] = vector2<float>{ static_cast<float>((rand() % 100)+50) };
            ++index;
        }
    }
}
```

   i. First of all, I initialize translations, colors, and scales value for instance.

```
std::map<std::string, vector2<float>*>    arrayVector2Uniforms{};
std::map<std::string, vector3<float>*>    arrayVector3Uniforms{};
```

ii. After then, I add two type of map which can store an array of vector2(translation, scale) and vector3(color).

iii. In addition, I should implement the send uniform function to send these new type of data.

```
void Graphics::Shader::SendUniformVariable(const std::string& variable_name, vector2<float>* number) noexcept
{
    glCheck(glUniform2fv(GetUniformLocation(variable_name), static_cast<int>(Sketch::max_size_instancing), &number->elements[0]));
}

void Graphics::Shader::SendUniformVariable(const std::string& variable_name, vector3<float>* number) noexcept
{
    glCheck(glUniform3fv(GetUniformLocation(variable_name), static_cast<int>(Sketch::max_size_instancing), &number->elements[0]));
}
```

iv. Thus, I add two more overloaded SendUniformVariable

```
for (const auto& element : material.arrayVector2Uniforms)
{
    material.shader->SendUniformVariable(element.first, element.second);
}
for (const auto& element : material.arrayVector3Uniforms)
{
    material.shader->SendUniformVariable(element.first, element.second);
}
```

v. and use it when uniform values should be transferred.

```
glCheck(glDrawArraysInstanced(vertices.GetVerticesListPattern(), 0, vertices.GetVerticesCount(), instanceCount));
```

vi. Lastly, if we just call of glDrawArraysInstanced, everything works.

vii. It draw 100 of object with only one call with given translation, color, size value of each object.