Sami Emre Erdogan
2019280513

Assignment4 - Simple particle effect: Snowing

Since this assignment was based on Tutorial7 I have used it as my refence base code and built my program on top of it. The first I have done is that I had to modify the ParticleSystem.h to create a class called snowflake to initialize the necessary functionalities and properties of a snowflake such as gravity, speed force, size position etc. of course I had to decide which properties had to be private or public.

Example code:

```
class SnowFlake
{
    glm::vec3 gravAcceleratio;
    glm::vec3 initialSpeed;
    double initialLife;
    glm::vec3 force;
    glm::vec3 speed;
    double mass;
    double life;

public:
    double scale = 10.0f;
    glm::vec3 position;
    glm::vec4 color;

public:
    SnowFlake(double m, glm::vec3 positionValue, glm::vec3 speedValue,
              glm::vec3 gravity, double lifeValue, double scaleValue)
    {
        gravAcceleratio = gravity;
        position = positionValue;
        speed = speedValue;
        initialSpeed = speedValue;
        scale = scaleValue;
        color = glm::vec4( _x: 1.0f,  _y: 1.0f,  _z: 1.0f,  _w: 0.5f);
        life = lifeValue;
        initialLife = lifeValue;
        mass = m;
        force = glm::vec3( _x: 0.0,  _y: 0.0,  _z: 0.0);
    }
```

The assignment wanted us to use different size so I have used the rand() function within the width so each time it would display a different size of snowflake. Of this will be done in an increasing while loop.

```
scaleValue: 10.0f + rand() % 60));
```

Sami Emre Erdogan
2019280513

Also I had to create an array to setup the mesh and contain the attribute properties for the particles and the background image.

```
GLuint backgroundVBO, backgroundVAO;
GLfloat background_quad[] = {
        0.0f, 1.0f, 0.0f, 1.0f,
        1.5f, 0.0f, 1.0f, 0.0f,
        0.0f, 0.0f, 0.0f, 0.0f,

        0.0f, 1.0f, 0.0f, 1.0f,
        1.5f, 1.0f, 1.0f, 1.0f,
        1.5f, 0.0f, 1.0f, 0.0f
};
// Set up mesh and attribute properties
GLuint VBO, VAO;
GLfloat particle_quad[] = {

        0.0f, 1.0f, 0.0f, 1.0f,
        1.0f, 1.0f, 1.0f, 1.0f,
        1.0f, 0.0f, 1.0f, 0.0f,

        0.0f, 1.0f, 0.0f, 1.0f,
        1.0f, 0.0f, 1.0f, 0.0f,
        0.0f, 0.0f, 0.0f, 0.0f,
```

Lastly every second passed in the program I had to make sure that the snowflake had to increase divisible by 200 then increase the snowflake.

```
if (rand() % (200) == 0) {
    sf_num++;
    snowflakes.push_back(SnowFlake( m: 0.5,  positionValue: glm::vec3( _x: rand()
                                     speedValue: glm::vec3( _x: 0.0,  _y: rand()
```