In my project, you are inside of room with wooden wall and frame for window behind, and you enjoy view of snow falling and forest of three trees. Do not worry the window is closed.

I did my project in four main steps

1. Implementation of Snowflake
2. Animation
   1. Falling
   2. Rotation and waving
3. Drawing Gradient Rectangle
4. The scene
   1. Room
   2. Outside

1. The snowflake.

Firstly, I start with defining base triangle, from which I am going to grow fractal. I chose 2 points on the plane and third defined as follow

Where M is rotation by 60 degree. Therefore, I get my equilateral triangle.

Next step is to create Queue and input every side of initial triangle in it. From this point, I start to iterate until Queue is empty.

* Pop pair of points which describes side
* Compute new point x1 and x2, so that they trisect side.
* Rotate to find x3 as in above
* Add 4 new sides in queue

Stopping condition is passed down with pair of points as Generation, whenever new side are added to Queue in step 4, Generation increments by 1 and initial triangle was generation 0. I choose 6th generation as last.

When template is ready, I scaled snowflakes to small size, gave them random color and put them in random places above clipping volume.

Animation

For smooth motion I need glwfGetTime function, my argument for DeltaTime is how much time passed since last loop.

Easiest thing to do is define falling motion; it is just decreasing y-component by some value. However, if everything falls with the same speed it is not interesting, so I define additional variable Slow, and every snowflake by DeltaTime/Slow units. After moving snowflake, I increase value of slow so that next will be a little bit slower, I could do the same with std::rand(), however I like this approach.

Next step is to make them rotate and oscillate. Rotation was easy to do just change orientation of snowflake. As for oscillations, I used cos, and used as an argument glwfGetTime() – count, so that they will be asynchronous.

Drawing gradient rectangle

First, I need mesh to draw on, it easy to define it with four vertices and GL\_TRIANGLE\_STRIP. Secondly, I need to implement shader. I chose 2 colors which I think appropriate to depict sky, and write simple linear function to steadily change from 1 to another.

Scene

One of first thing I had to do is make frame. Frame is just rectangle with hole inside. So I need some set of points to depict it. If I have chosen to use GL\_TRIANGLE mode, I would need at least 8 triangles or 24 vertices. Instead, I decided to use GL\_TRIANGLE\_STRIP. Now I need to triangulate it in some nice manner, after thinking about it, I choose to do it this way

It is very nice, because requires only border points to describe, and 9 points to define all triangles. Moreover, function can be implemented easily and general.

Next step is just define frame and walls, using above function. Now we need to put glass in window frame. It is simple rectangle and have to be transparent; unfortunately, we cannot make it transparent, so I resorted to use GL\_BLEND mode.

Finally outside, first we need ground it is simple whitish rectangle. After I decided to add some trees.

I defined three vertices for triangle top of tree, next step is to translate it on y-coordinate 2 times to make tree-like figure and add rectangular stump.

My Homework project satisfies all requirements,

* I made fractal snowflake
* the animation of movement is natural and smooth
* I used 2 types of primitives to generate background objects, for example for tree I used both triangle strips and triangles
* Gradient rectangle switches from light blue to whitish color with altitude

I think my project have some interesting features like deciding speed of snowflakes so all of them asynchronous, I have not only background objects but also foreground object Room, all object are at different depth so that perspective doesn’t look weird. They way the triangulation of rectangle with hole is made is also fascinating