The first part of my system that I created are the two trees. I made a class to generate and draw these trees based on given degrees, levels, rules, and an axiom.

Both trees use the same alphabet: “F” = Forward, “X” = Placeholder, “R” = Reduce branch, “E” = Expand branch, “-” = Rotate negative, “+” = Rotate positive, “B” = Berry/Circle Drawn, “0” = Leaf 0, “1” = Leaf 1, “[“ = Push matrix, “]” = Pop matrix

Both trees use the same axiom: “X”

Tree 1 rules: “X” -> “RF-[[X[-1]]+XB]+F[+FX[+0]]-XE”, “F” -> “FF”

Tree 2 rules: “X” -> “RF-[[X[-0]]+XB]+F[-FX]+F[-FX[+1]]+XE”, “F” -> “FF”

I realized after that I may have made the rules too complex, and ideally I would break down these rules in the future to be in simpler steps. However, I really wanted to make sure there was enough variation between the two trees to give the illusion of a kind of natural forest. I chose muted colors to make it appear slightly wintery.

The second part of my system that I created are the snowflake systems. I made a class like the Trees to generate these snowflakes that I had originally created for my FIT3. Despite the fact that the snowflakes don’t look entirely natural, I liked the unique form I came up with since it looks much more intricate than other examples I had seen in class. I used two different colors to add more detail to the pattern. I also made them way larger than life so that the detail and system is visible. .

Snowflakes have a limited alphabet: “G” = Forward, “+” = Rotate positive, “-” = Rotate negative, “[“ = Push matrix, “]” = Pop matrix

Snowflake axiom: “G+G+G+G+G+G+G+G”

Snowflake rules: “G” -> “GG[--G][++G]”

As for extensions, I implemented bezier curves on the tree to create the leaves. I created two different Bezier curves to use, so there would be variation in the leaf structure on both of the trees. These Bezier curves are pretty minute, however I think they add to the variation to make the trees appear slightly more natural and less… structured. For another extension, I implemented a particle-like system for drawing the snowflakes. This is so incredibly basic, but each snowflake has a position and falling velocity, along with behavior for when they hit the ground. I made it so that the snowflakes fall from the sky and then respawn at a random location at the top when they hit the bottom, instead of creating new snowflakes (since that’s really memory-intensive). Further, I added some interaction to the snowflakes. The snowflakes will slowly follow where the mouse is, as if the mouse is mimicking wind. This is at a slower rate than their falling. Finally, I added a kind of breathing feature to the trees. The degrees for the angle will vary slightly as you move your mouse left to right. I originally wanted the trees to bend left or right as if the wind was blowing the trees along with the snowflakes, however I found that a little too difficult to implement for this project.