Justin Dietrich

Ryan Scott

9/11/2022

CST – 310

Prof. Citro

Sierpinski Triangle

**In this project, we are to understand how to make a Sierpinski Triangle using OpenGL and C++.**

Example code for a 2D and 3D triangle were provided, so the majority of this project will go to explaining how it works. Reading the code comments and testing different values gave enough of an understanding of what is going on.

Both the 2D and 3D triangle code work mostly the same. It’s just that the 3D code adds a z coordinate to the points. First, the code establishes the coordinates of the three/four vertexes of the initial triangle/pyramid. Starting at the first vertex, a random vertex of the initial triangle is chosen, and a dot is drawn exactly between the current point and the randomly chosen point. With this newly drawn dot as the current point, another random initial vertex is chosen and another point is drawn perfectly in the middle of the two points again. The process repeats until the established point limit is reached.

Diagram

Description automatically generated

Some extra code that the 3D version has sets the points to be colored by their z value, giving a quick 3D perspective effect.

Here are screenshots of the 2D and 3D code, as well as an edit I made of the 2D code.

Shape

Description automatically generatedArrow

Description automatically generatedBackground pattern

Description automatically generated

I edited the code so that the initial points were placed differently, and the point limit (k) was reduced from 100 000 to 1 000. Here is a screenshot of the portion I edited.

Graphical user interface, text, application

Description automatically generated

* ‘k’ changed from 100 000 to 1 000
* Points in vertices[] coordinates were changed.