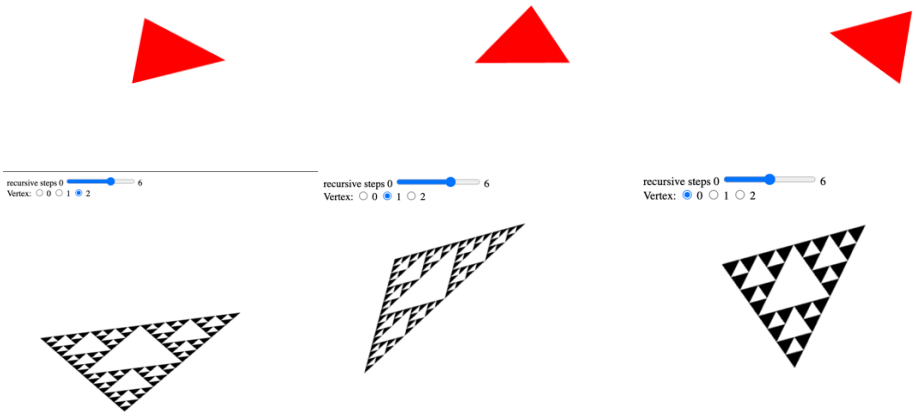
Lab 2.2 : Interaction and Animation:

1) **Paste all screenshots here**



2) **What is the purpose of gl_Position.w?**

gl_Position.w is the homogeneous coordinate and tells us wether the element is a vector or a point.

3) **What happened after you added the initial code to triangle's render function and what do you have to do to fix it?**

The triangle flew off into the right corner and off the canvas. We need to prevent the triangle's vertices from having values outside the canvas (must be less than 1). Basically we need to add collisions with the sides of the canvas.

4) **Assuming everything works fine with your animated moving triangle, add a delay so that it doesn't update as quickly as possible. Add a 100ms delay between each call to render. Copy and paste your code below. (Hint, see the slide titled 'Adding an Interval').**

```
function render() {
        setTimeout(function() {
                gl.clear(gl.COLOR_BUFFER_BIT);
                theta += 0.1;
                gl.uniform1f(thetaLoc, theta);
                gl.drawArrays(gl.TRIANGLES, 0, 3);
                window.requestAnimFrame(render);
        }, 100);
}
```

5) **Answer the question regarding console.log from the Sierpinski Clicker section.**

   The coordinates that are being displayed start in the top left and range from 0 to the canvas' width/height, while WebGL uses a coordinate system that starts in the middle of the screen and range from 1 to -1.

6) **Answer the question regarding the math surrounding the mouse click event.**

   We have to change coordinate spaces to use WebGL's coordinate space. As described in the previous answer, the coordinates of the mouse's position is relative to the top left of the canvas, but we need the coordinates relative to the center.

7) **Extend the Sierpinski in some fashion. Describe your extension and how you had to modify the original code to get it.**

   I made the triangle bounce around the frame, similarly to how we animated the triangle in triangle-anim.html/js. I also added the clicking element from sierpinski-click.html/js. Both of these features do not overlap in their implementation so It was relatively easy to bring both together.

8) **Copy and paste the URL of your homepage (just in case if it has changed)**

   https://student.computing.gvsu.edu/johnsev/cis-367.html

9) **Time to start thinking about your term project. For this, briefly describe *two* ideas of what you would like to do. *Note: you are not going to be held to these, but you will get feedback on them!***

   a) I've created a Ray Caster in pygame in the past and had a lot of fun doing it. I think it could be interesting creating one in WebGL and JavaScript as I have no experience working in JavaScript or HTML. I'm not sure If I really want to do this yet though, but It's still on the board.

   b) I really liked the fluid simulation shown at the beginning of class. I've always been interested in different types of simulations so it might be interesting to look into creating something similar. I don't have much experience in unity or blender so I'd love to spend more time in those programs.