**Kaitlyn Zahn**

***Assignment 3 Report***

**February 18, 2021**

**\_\_**

**CS 4610 Computer Graphics 1**

**\_\_**

**Ye Duan**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

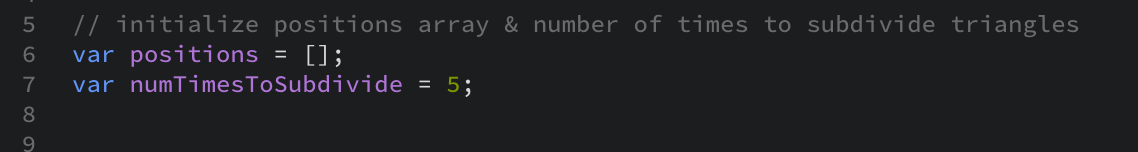
**Link to YouTube Video:**

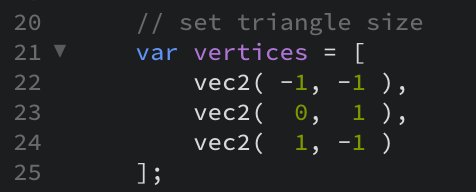
https://youtu.be/47Py9QCF9N8

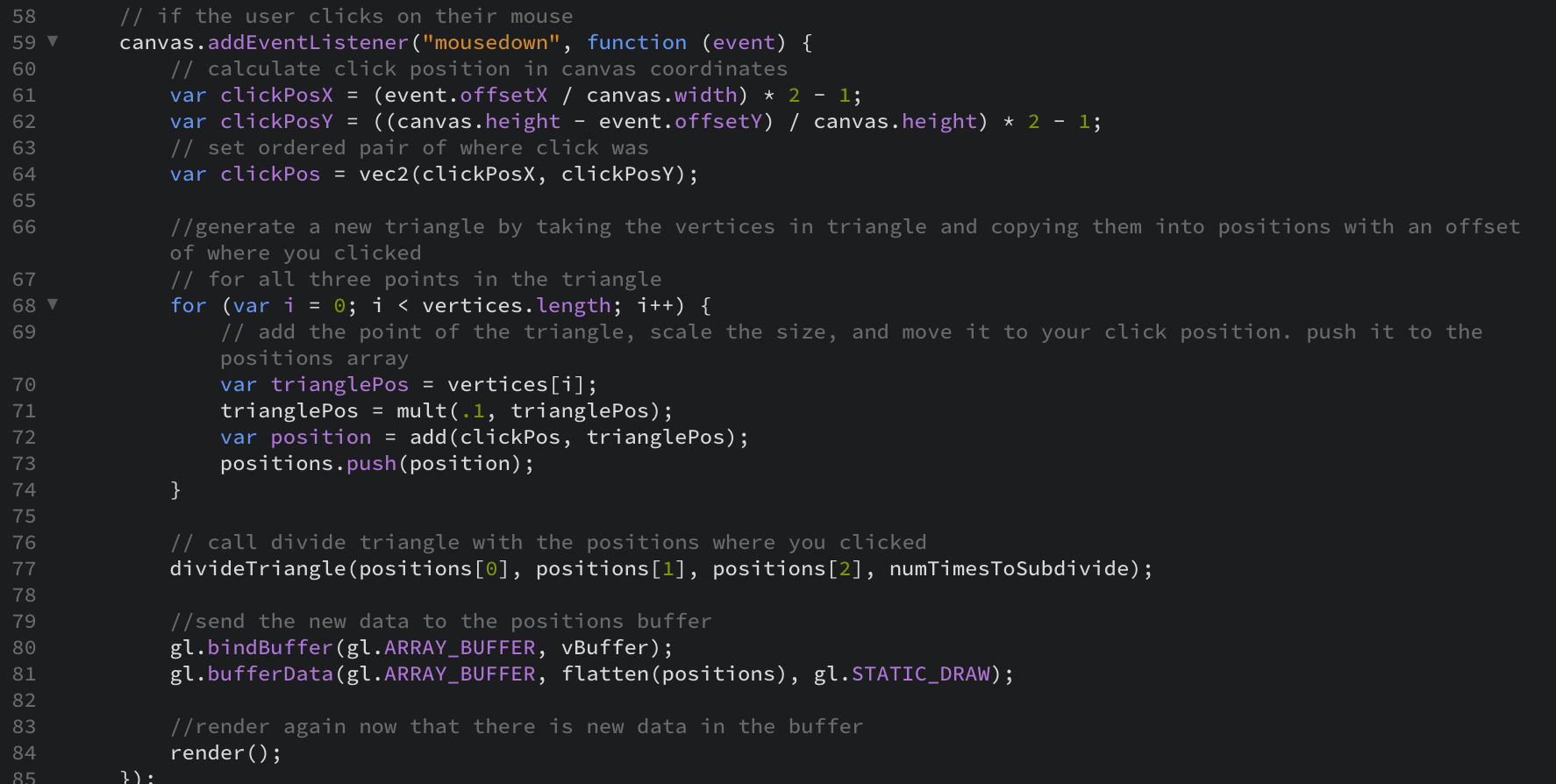
**Part A**: Runtime Vertex Manipulation (5 points)

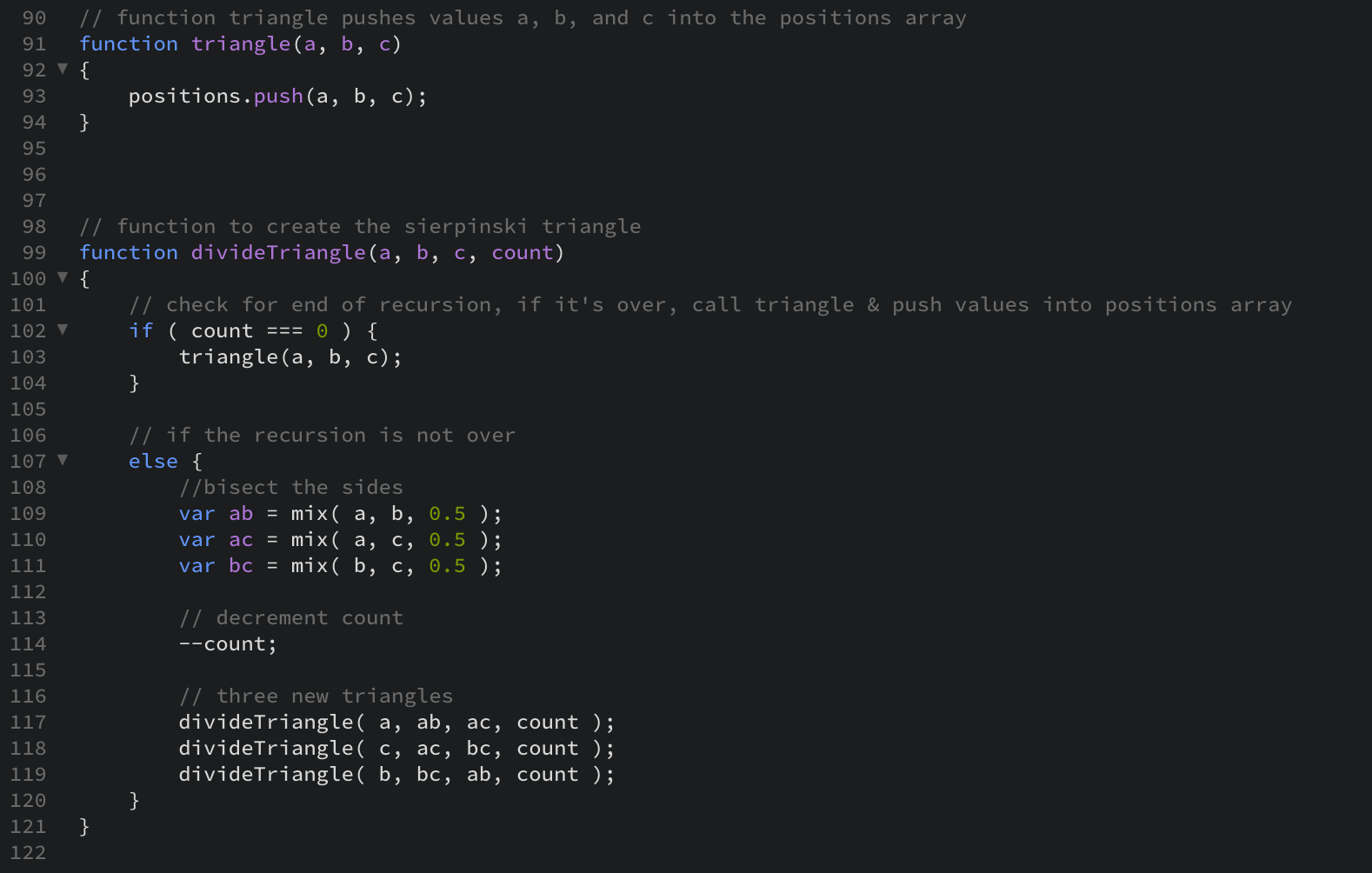
*Combine DrawSquare and SierpinskiGasket to draw a sierpinksi triangle at the mouse click position*

This code moves the positions array to be a global array, creates triangle vertices, and then uses a mousedown event listener to take the position of the click, change the size of the triangle, add an offset, and push the click location to the positions array. Then, it calls divideTriangle from SierpinskiGasket to render the triangles.

**

**

**

****

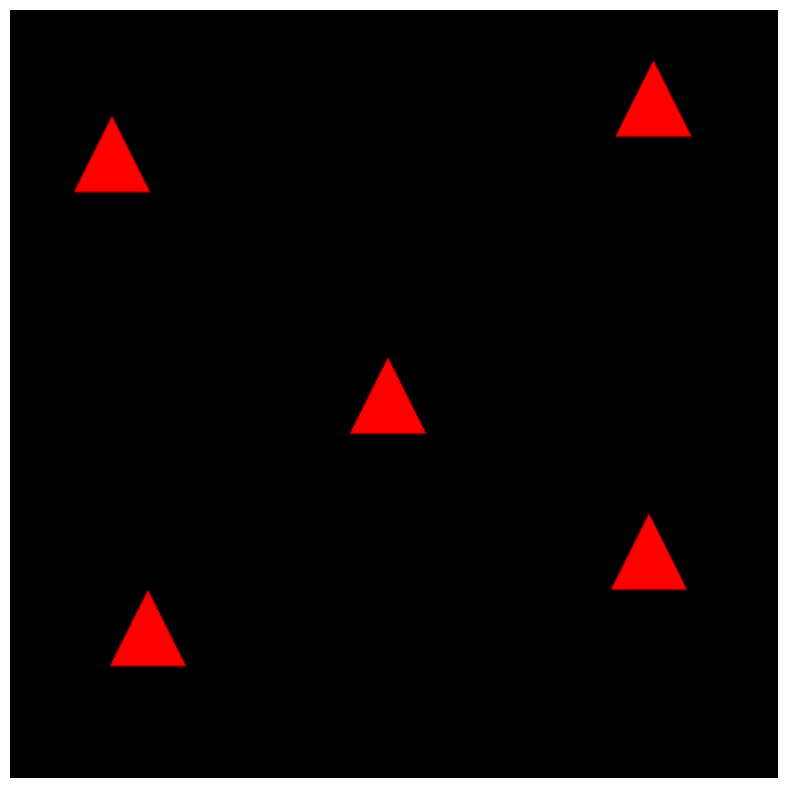
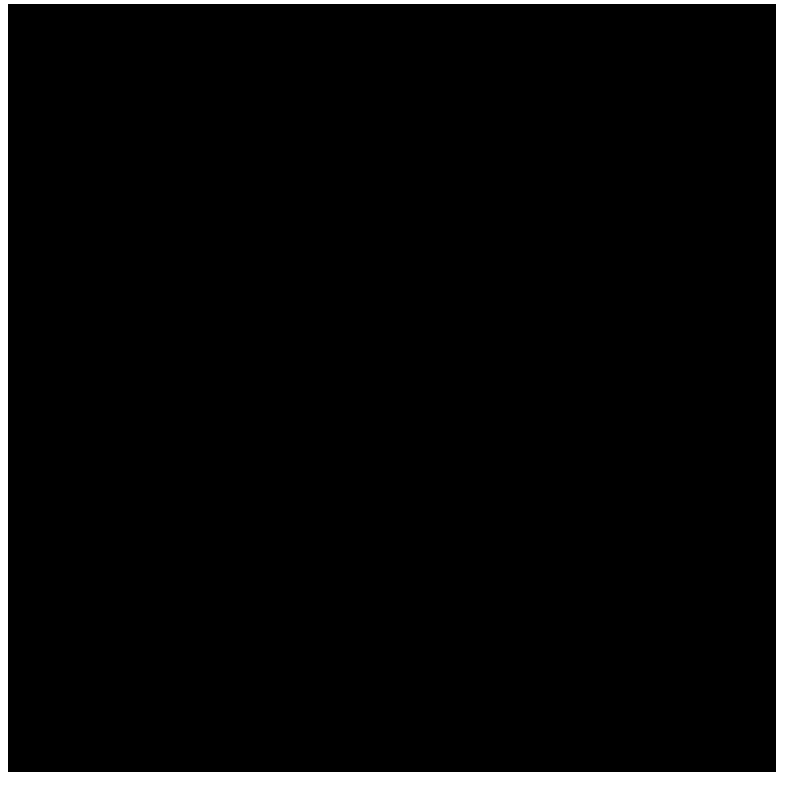
******

Figure A.1.2 SierpinskiGasket & DrawSquare

Figure A.1.1 SierpinskiGasket & DrawSquare Original

Note: I could not figure out how to get the SierpinskiGasket triangles on click.

**Part B**: Model Matrix (5 points)

*Download the program InteractiveCube that will Render a 3D cube in WebGL.*

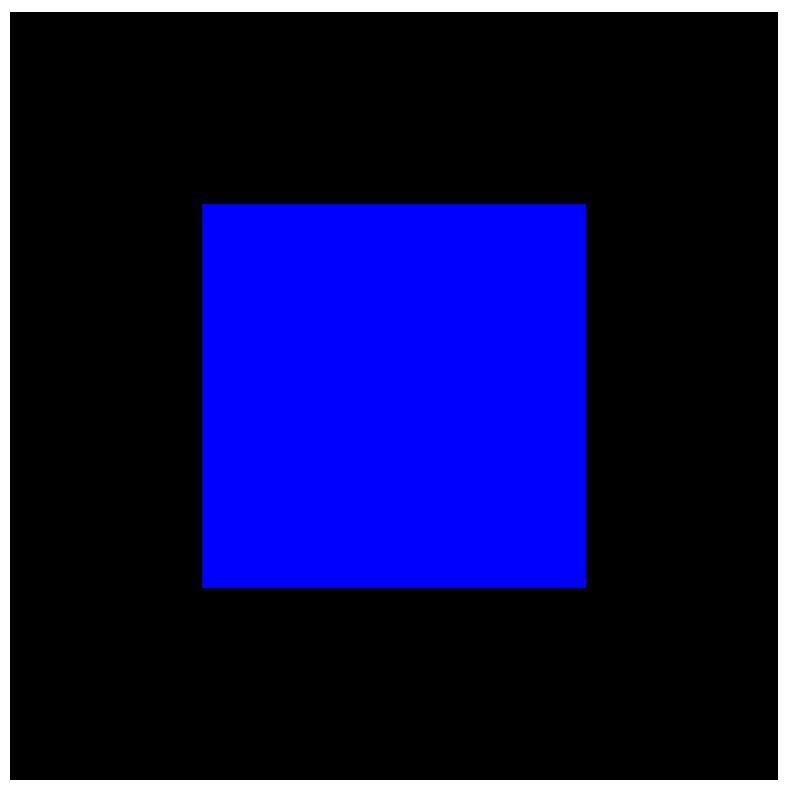
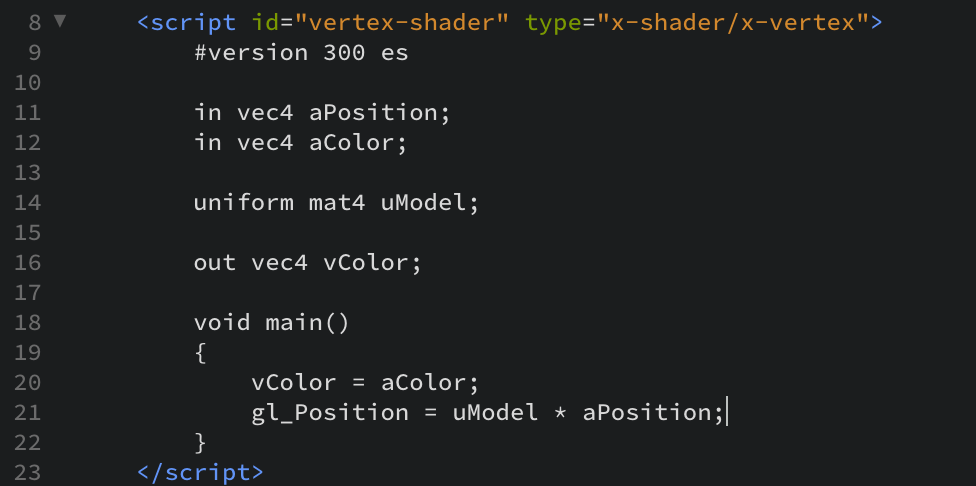
**

Figure B.1.1 Interactive Cube Original

*Hold W, A, S, and D, to move the cube up, left, down, and right. It should move smoothly and continuously while you continue to hold the key.*

This code sets uModel and gl\_Position in index.html. It adds a matrix uniform to the shader, sends the matrix to the shader, and uses KeyIsPressed to detect W, A, S, or D being pressed. In each case, it uses translation to move the cube.

**

**

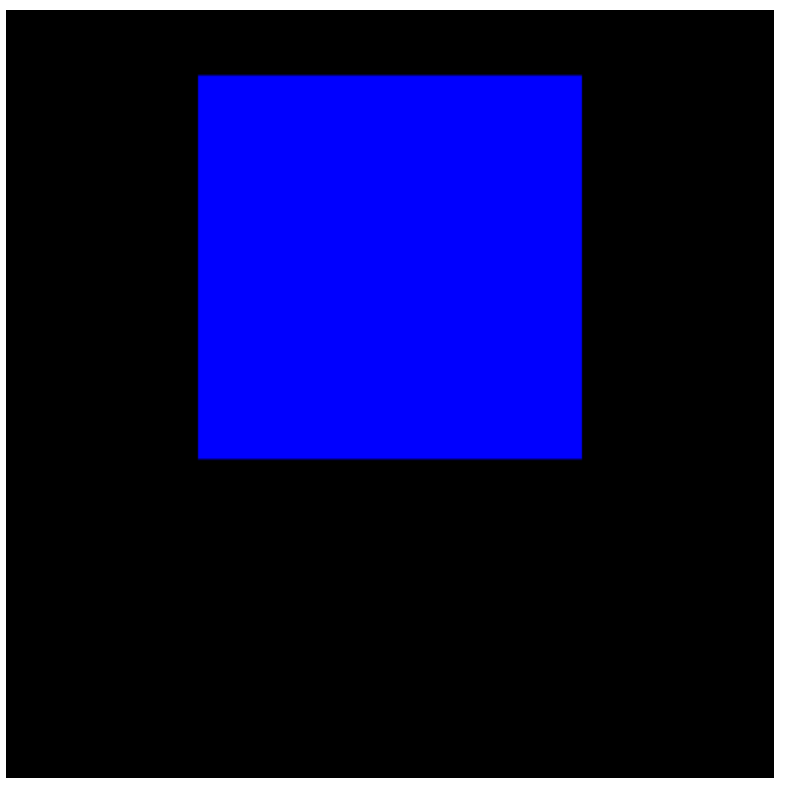
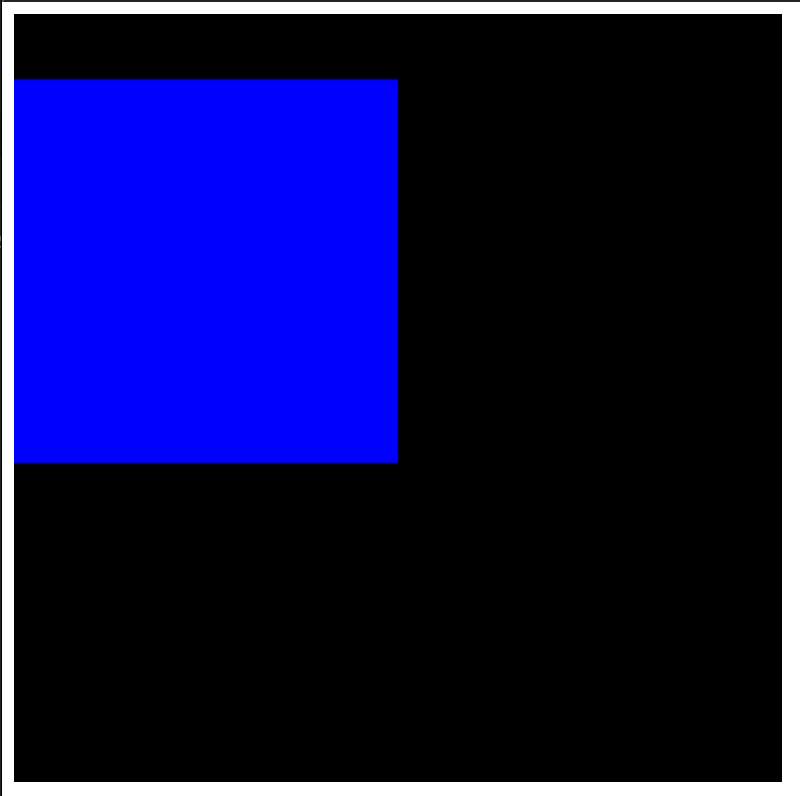
**

Figure B.2.2 Interactive Cube Moved Left

Figure B.2.1 Interactive Cube Moved Up

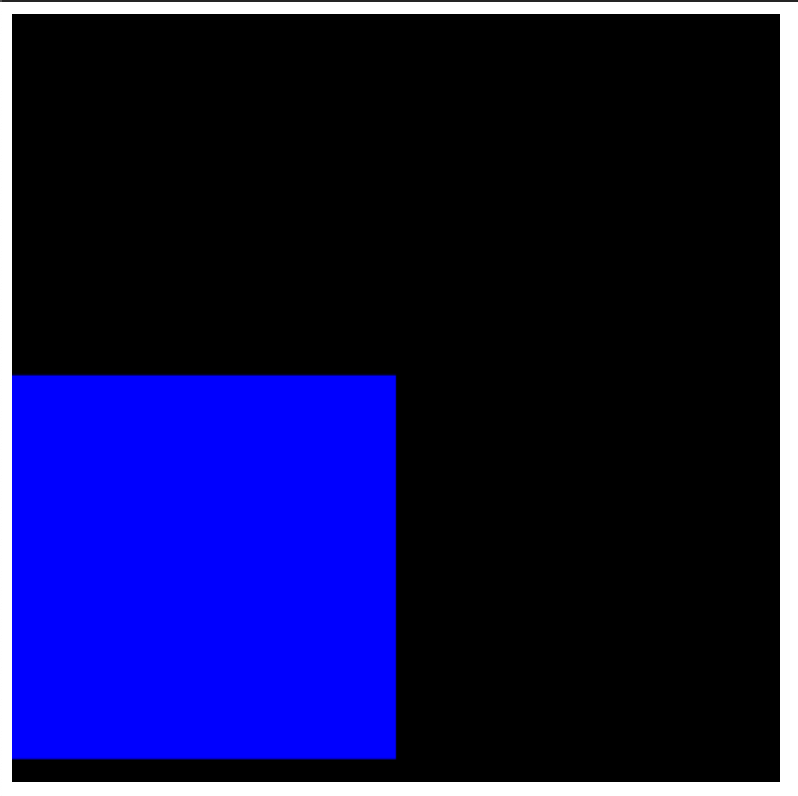
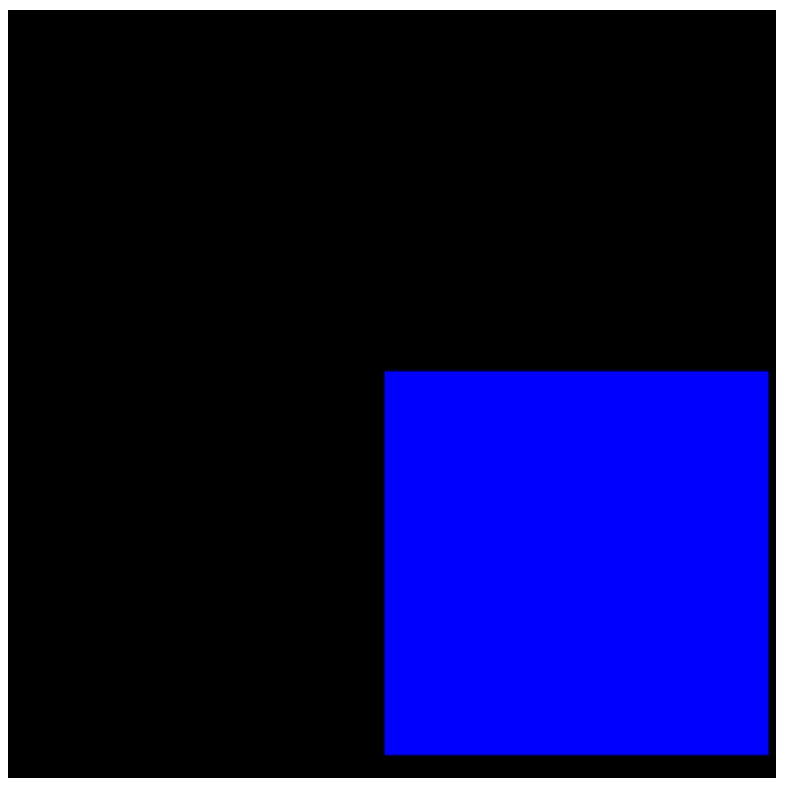
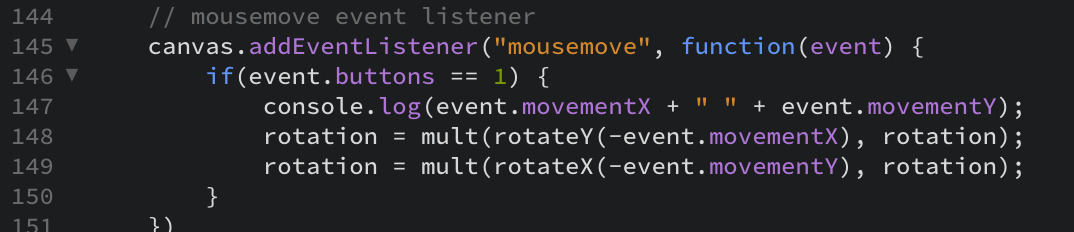
**

Figure B.2.4 Interactive Cube Moved Right

Figure B.2.3 Interactive Cube Moved Down

*Click and drag to rotate the cube*

This code uses the mousemove event, checks if the left mouse is pressed while the event fires, and then uses event.movementX and event.movementY to rotate the cube about the Y axis for X movement of the mouse and about the X axis for Y movement of the mouse.

****

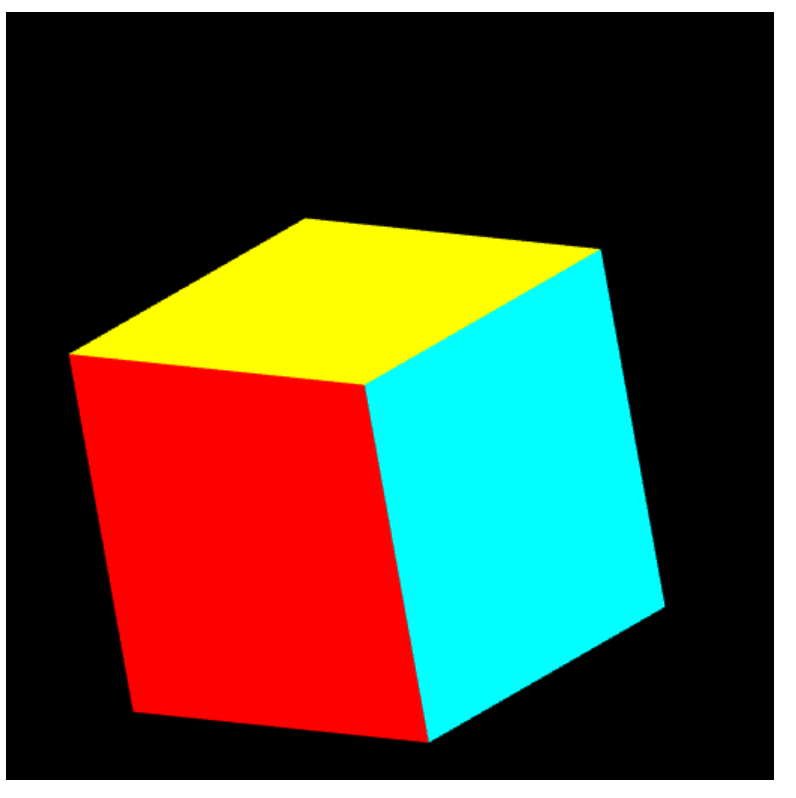
****

Figure B.3.3 Rotated Cube 3

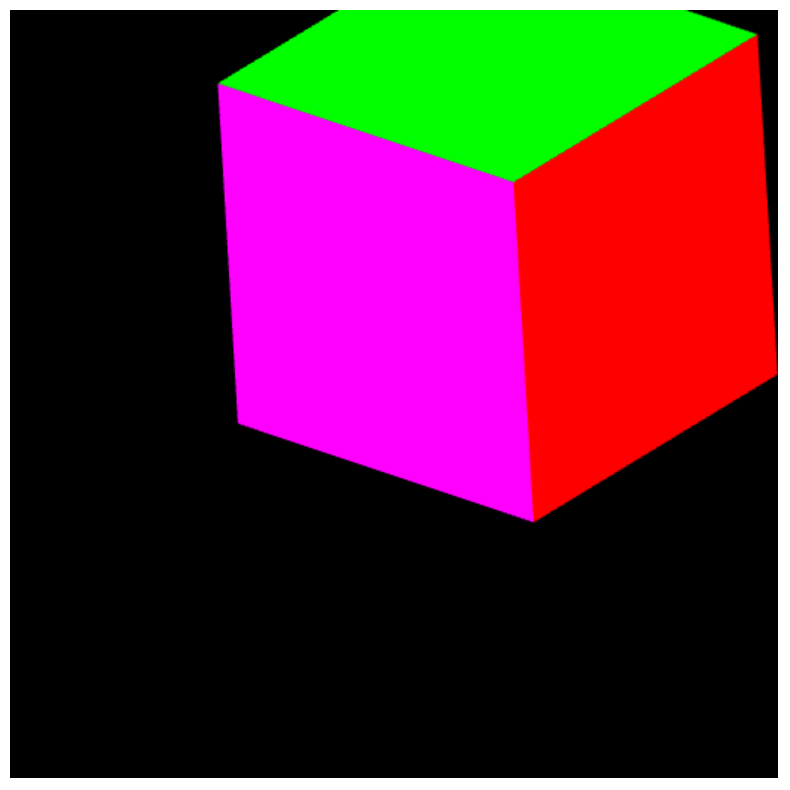
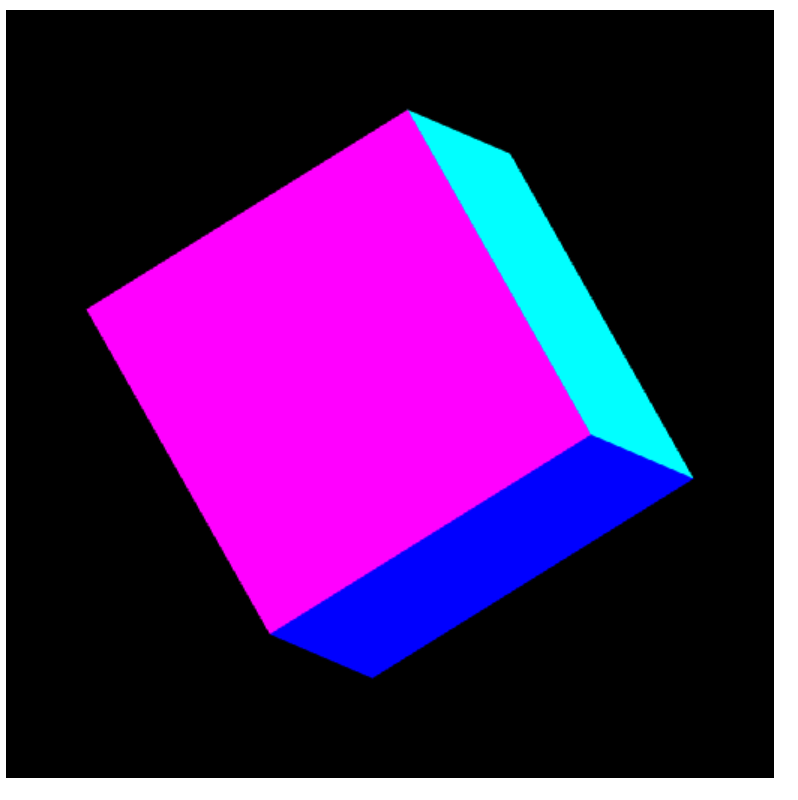
****

Figure B.3.1 Rotated Cube 1

Figure B.3.2 Rotated Cube 2

**Issues I Faced:**

I could not figure out how to get the Sierpinski Triangles rather than just a plain triangle on the mouse click. I could get it to load when I used vertices[0] rather than positions[0], but then I couldn’t place the triangle at the location of the click. I’m not sure what the error was for this, but I worked on it for a long time and was unable to solve this problem. The rest of Part A works, the triangle just isn’t dividing properly.