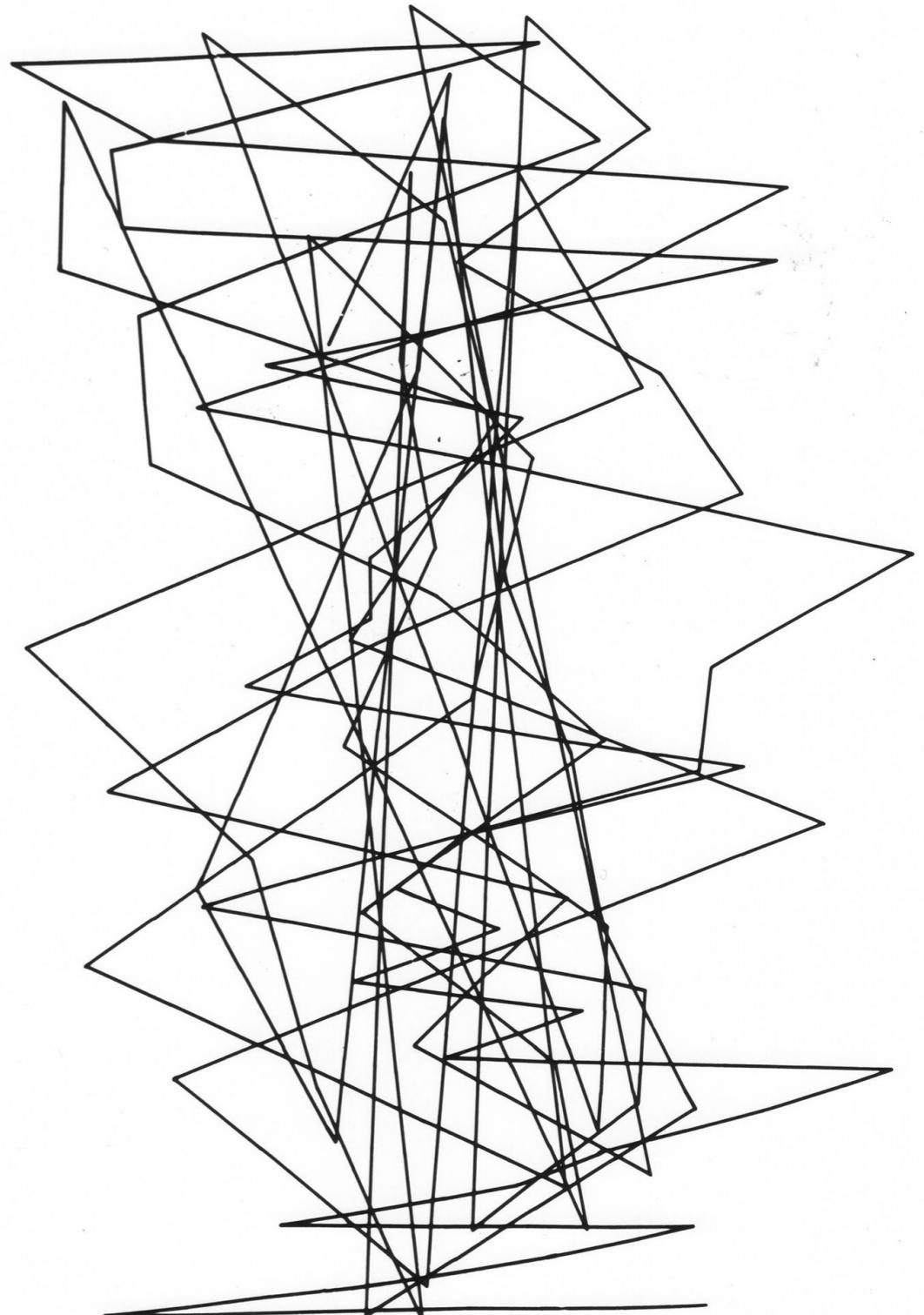


GETTING GENERATIVE WITH P5.JS

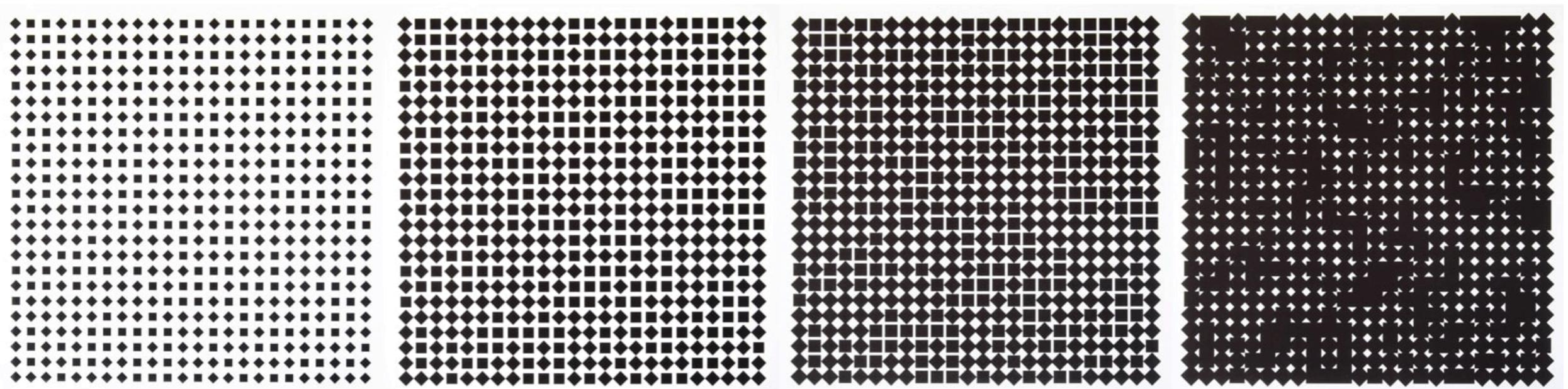
DANIEL HOWE
SHEUNG WAN, HK
FEBRUARY 16, 2019



GAUSSIAN - QUADRATIC (1963)
BY A. MICHAEL NOLL

SLIDES & CODE

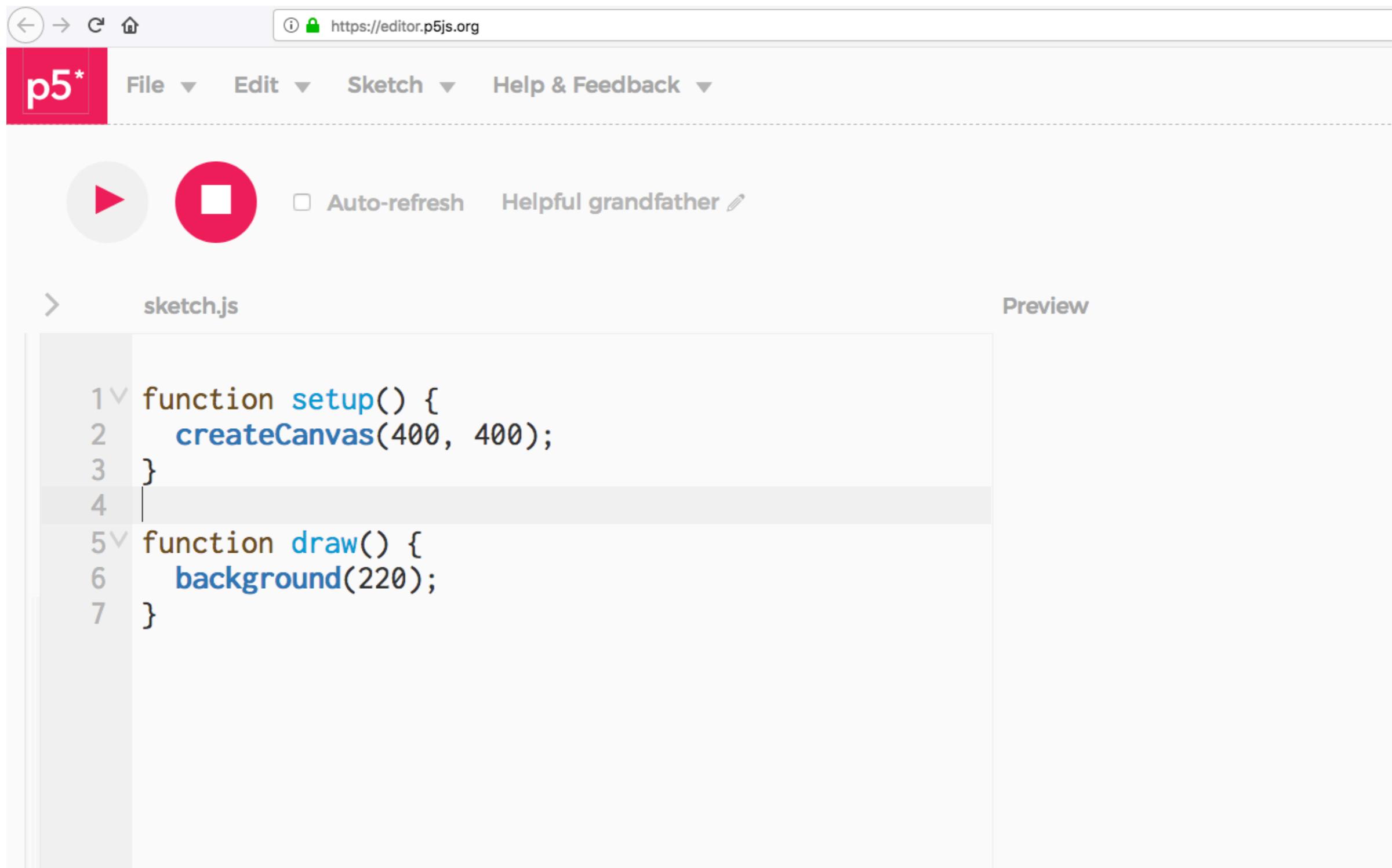
<https://github.com/dhowe/GetGen>



VERA MOLNAR, CARRÉS EN 2
POSITIONS 1/2/3/4 , 2011-13

**RECURSION
FRACTALS
AFFINE TRANSFORMS
RANDOM
NOISE**

TOOLS: P5.JS EDITOR



The screenshot shows the p5.js Editor interface. At the top, there's a browser-style header with back, forward, and refresh buttons, and a URL bar showing <https://editor.p5js.org>. Below the header is the p5 logo and a navigation menu with File, Edit, Sketch, and Help & Feedback. On the left side, there are two large circular buttons: a light gray one with a red play triangle and a red one with a white square. To the right of these buttons are checkboxes for "Auto-refresh" and "Helpful grandfather" with a pencil icon. The main area is titled "sketch.js" with a "Preview" button on the right. The code editor contains the following JavaScript code:

```
1< function setup() {
2   createCanvas(400, 400);
3 }
4
5< function draw() {
6   background(220);
7 }
```

TOOLS: REFERENCE

A screenshot of a web browser displaying the p5.js reference page. The address bar shows the URL <https://p5js.org/reference/>. Below the address bar, there is a horizontal navigation menu with links to "Processing", "p5.js", "Processing.py", "Processing for Android", and "Processing for Pi".



p5.js

Processing creativity times JavaScript dynamism

[Home](#)

Reference

[Download](#)

Can't find what you're looking for? You may want to check out [p5.dom](#) or [p5.sound](#).

[Start](#)

You can download an offline version of the reference [here](#).

[Reference](#)

[Color](#)

[Environment](#)

[Lights, Camera](#)

[Structure](#)

[Libraries](#)

[Constants](#)

[Events](#)

[Math](#)

[Transform](#)

[Learn](#)

[DOM](#)

[IO](#)

[Rendering](#)

[Typography](#)

[Data](#)

[Image](#)

[Shape](#)

[Examples](#)

[Books](#)

Color

[Community](#)

[Creating &](#)

[Setting](#)

[Reading](#)

[background\(\)](#)

[alpha\(\)](#)

[clear\(\)](#)

[blue\(\)](#)

[colorMode\(\)](#)

[brightness\(\)](#)

[fill\(\)](#)

[Search the API](#)

[WHEN] THE ARTIST USES A SYSTEM, SUCH AS A SET OF NATURAL LANGUAGE RULES, A COMPUTER PROGRAM, A MACHINE, OR OTHER PROCEDURAL INVENTION, WHICH IS SET INTO MOTION WITH SOME DEGREE OF AUTONOMY CONTRIBUTING TO OR RESULTING IN A COMPLETED WORK OF ART

GENERATIVE ART

RECURSION

IMAGE BY JAN MISSET
FOR DROSTE CACAO, 1904



re·cur·sion /ri-'kər-zhən/

n. When an entity is defined by one or more references to itself.

RECURSION

re·cur·sion

/ri-'kər-zhən/

n. See recursion.

RECURSION

EXAMPLE: FACTORIAL

$$1! = 1$$

$$2! = 2 * 1 = 2$$

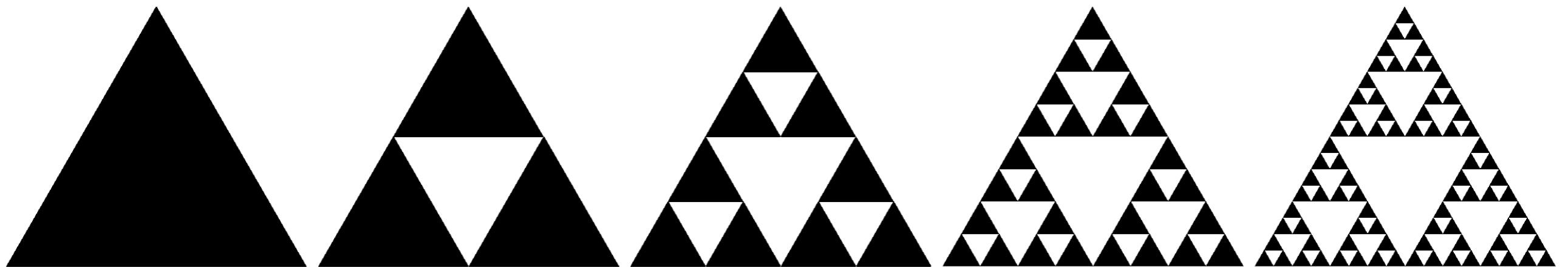
$$3! = 3 * 2 * 1 = 6$$

$$4! = 4 * 3 * 2 * 1 = 24$$

$$5! = 5 * 4 * 3 * 2 * 1 = 120$$

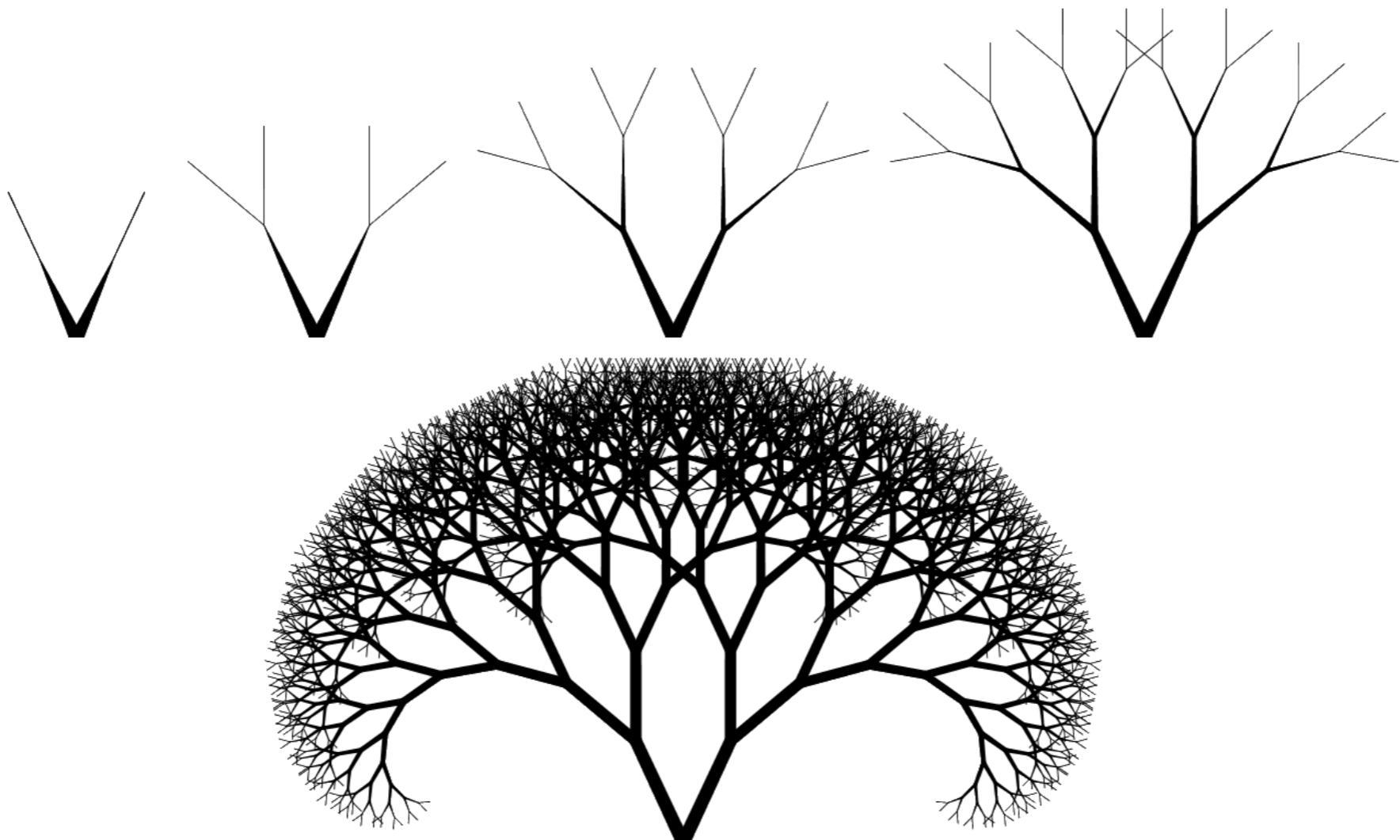
How can we define factorial() recursively ?

EXAMPLE: SIERPINSKI TRIANGLE



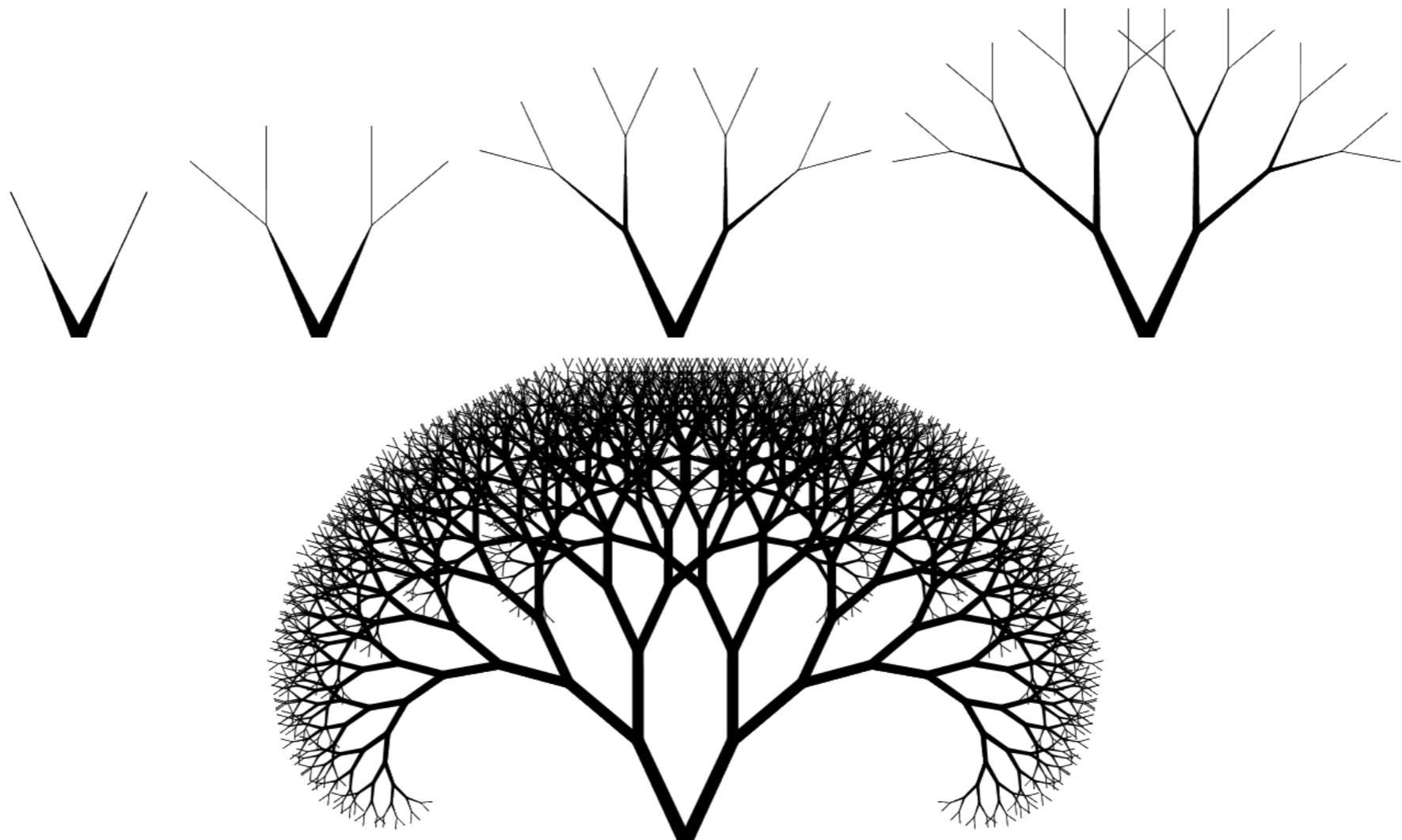
Make a new triangle from the 3 midpoints of the sides of the current triangle. Repeat for each of the three newly created triangles...

EXAMPLE: FRACTAL TREE



Draw a branch. From its end, rotate right and draw a shorter branch. Then rotate left and do the same. Repeat for the two new branches...

EXAMPLE: FRACTAL TREE



Draw a branch. From its end, rotate right and draw a shorter branch. Then rotate left and do the same. Repeat for the two new branches...

A NEW WAY TO DRAW SHAPES

There are two basic ways of drawing shapes in p5.js:

- 1) Position the shapes where we want them
- 2) Move the paper & draw shapes at the origin (0,0)

`translate()`

`rotate()`

`scale()`

AFFINE TRANSFORMATIONS

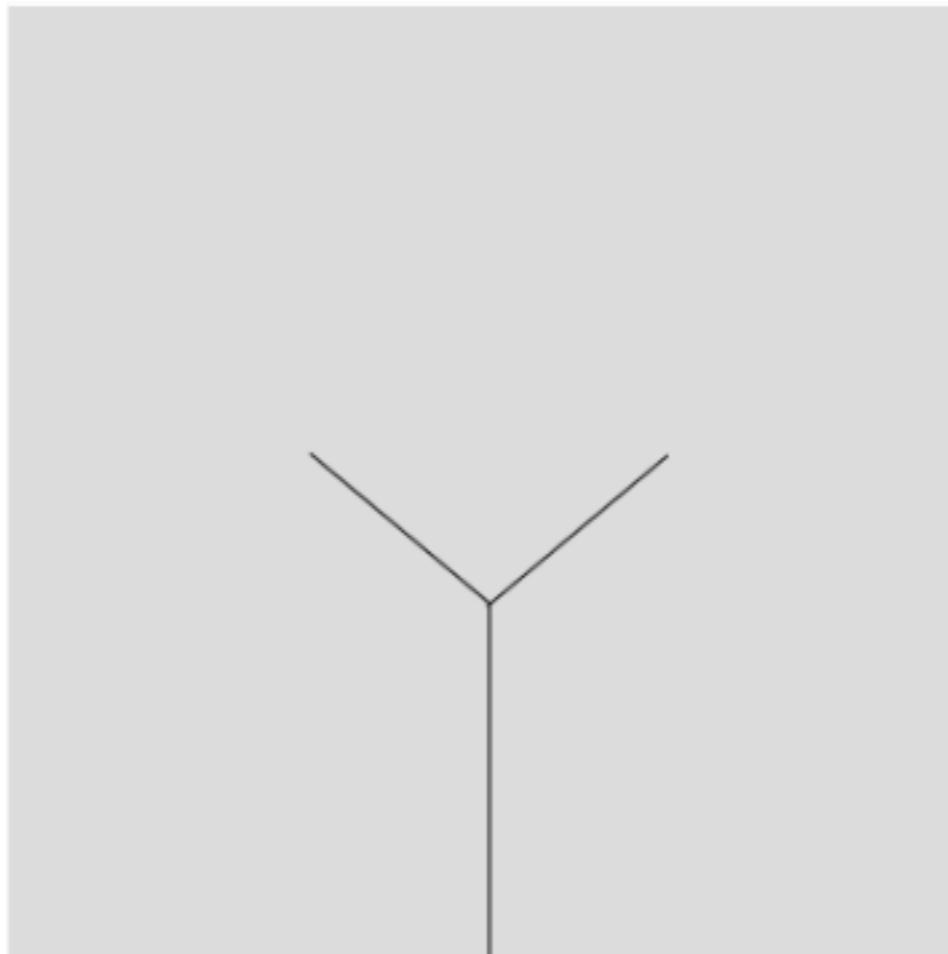
Functions that (essentially) move the 'graph paper':

`translate()` → move left/right or up/down

`rotate()` → rotate around the y-axis (2D)

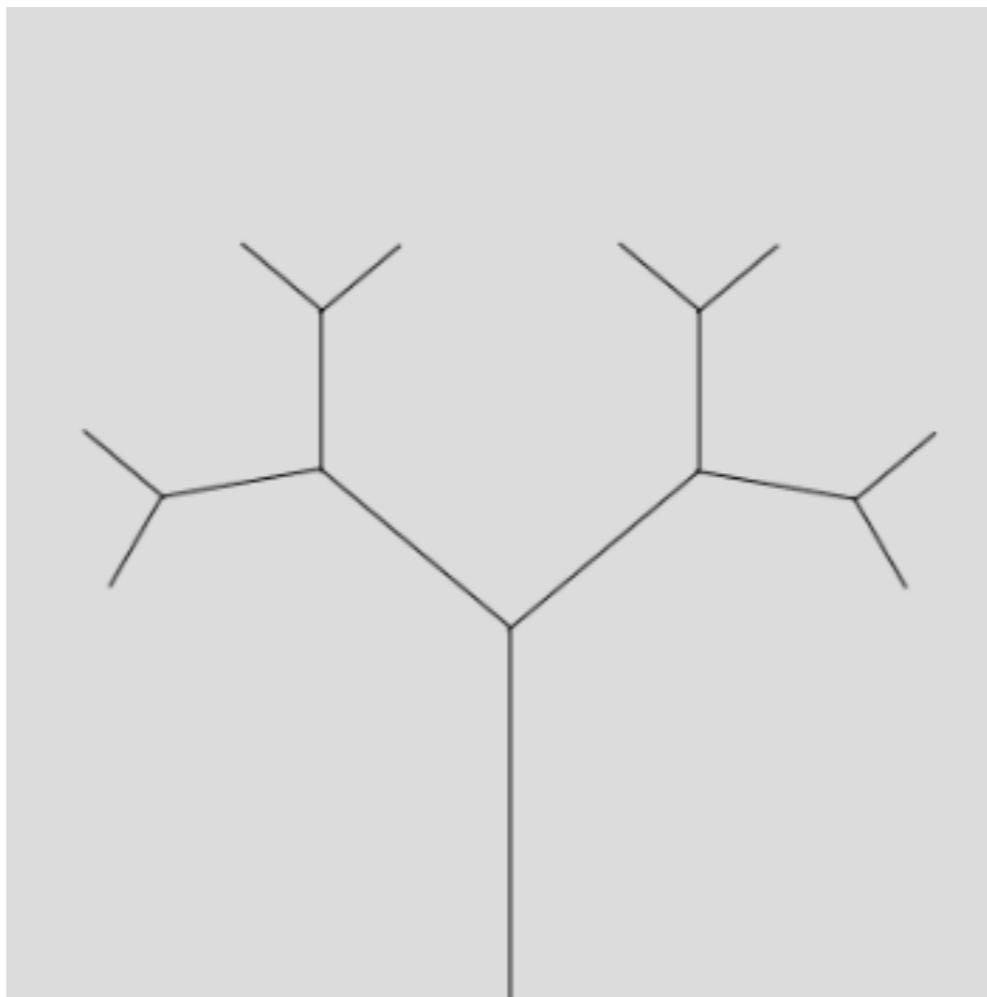
`scale()` → make bigger or smaller

FRACTAL TREE



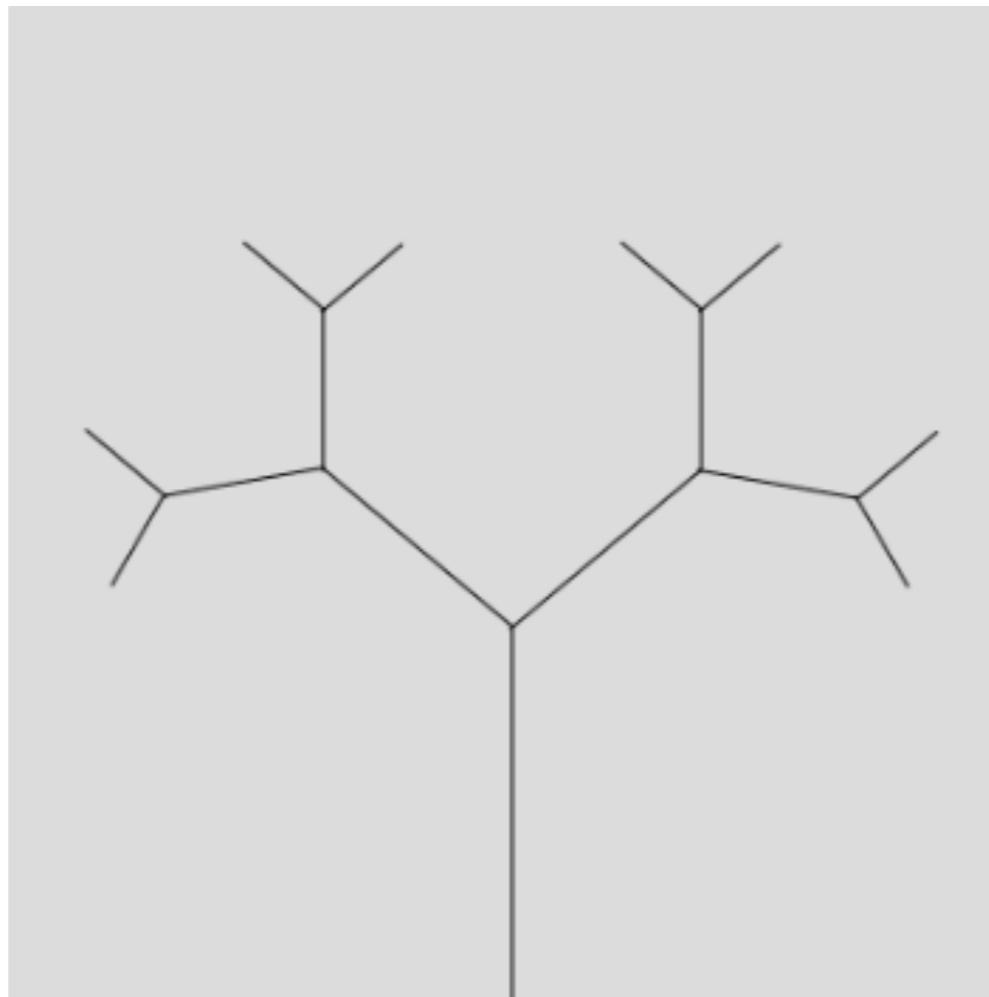
translate(), rotate(), line()

FRACTAL TREE



now lets experiment with our
variables: angle & lengthScale

MAPPING THE MOUSE POSITION



we can map() the angle to mouseX
and the lenScale to mouseY

THE MAP() FUNCTION

Description

Re-maps a number from one range to another.

map(value, start1, stop1, start2, stop2);

Parameters

value

Number: the incoming value to be converted

start1

Number: lower bound of the value's current range

stop1

Number: upper bound of the value's current range

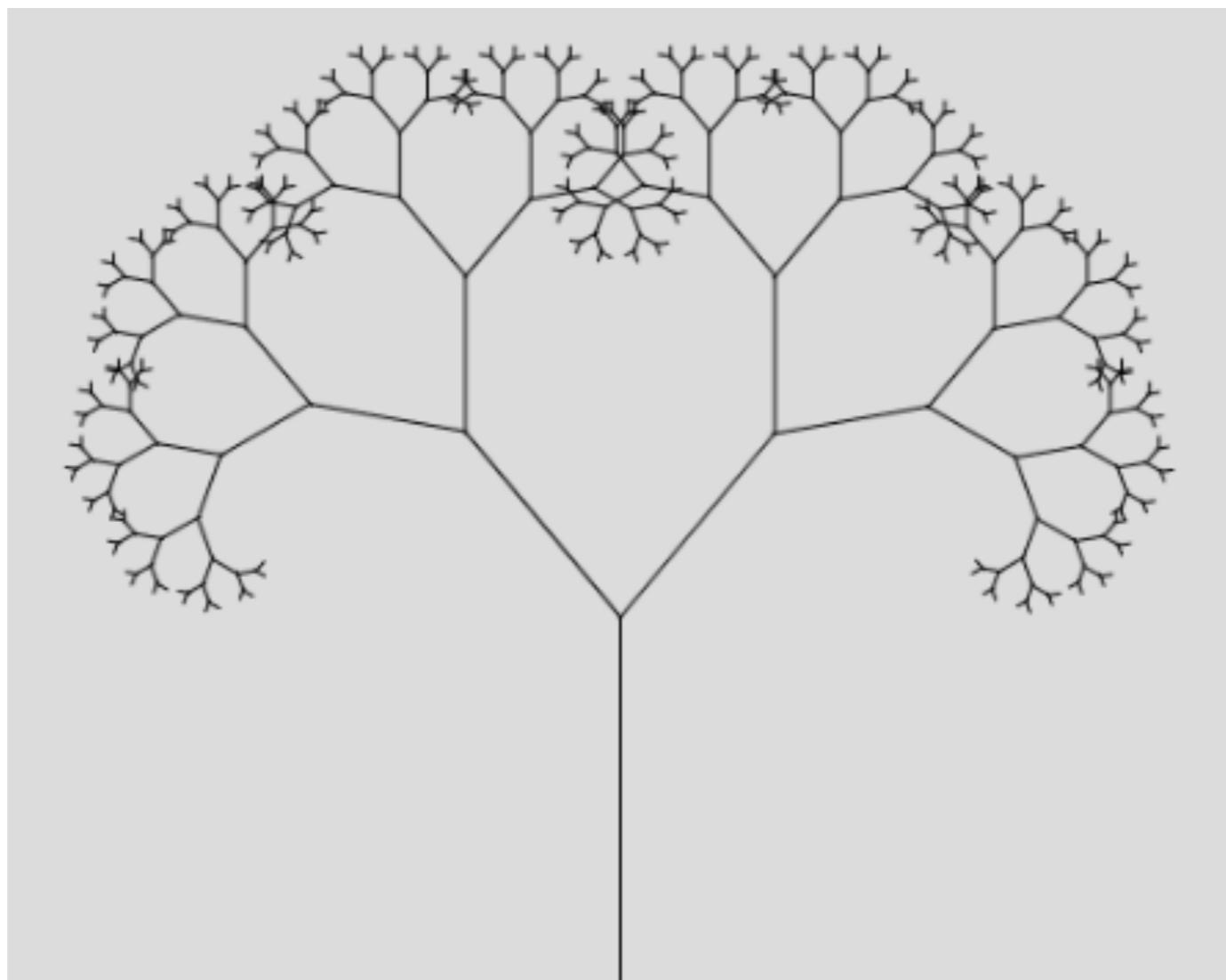
start2

Number: lower bound of the value's target range

stop2

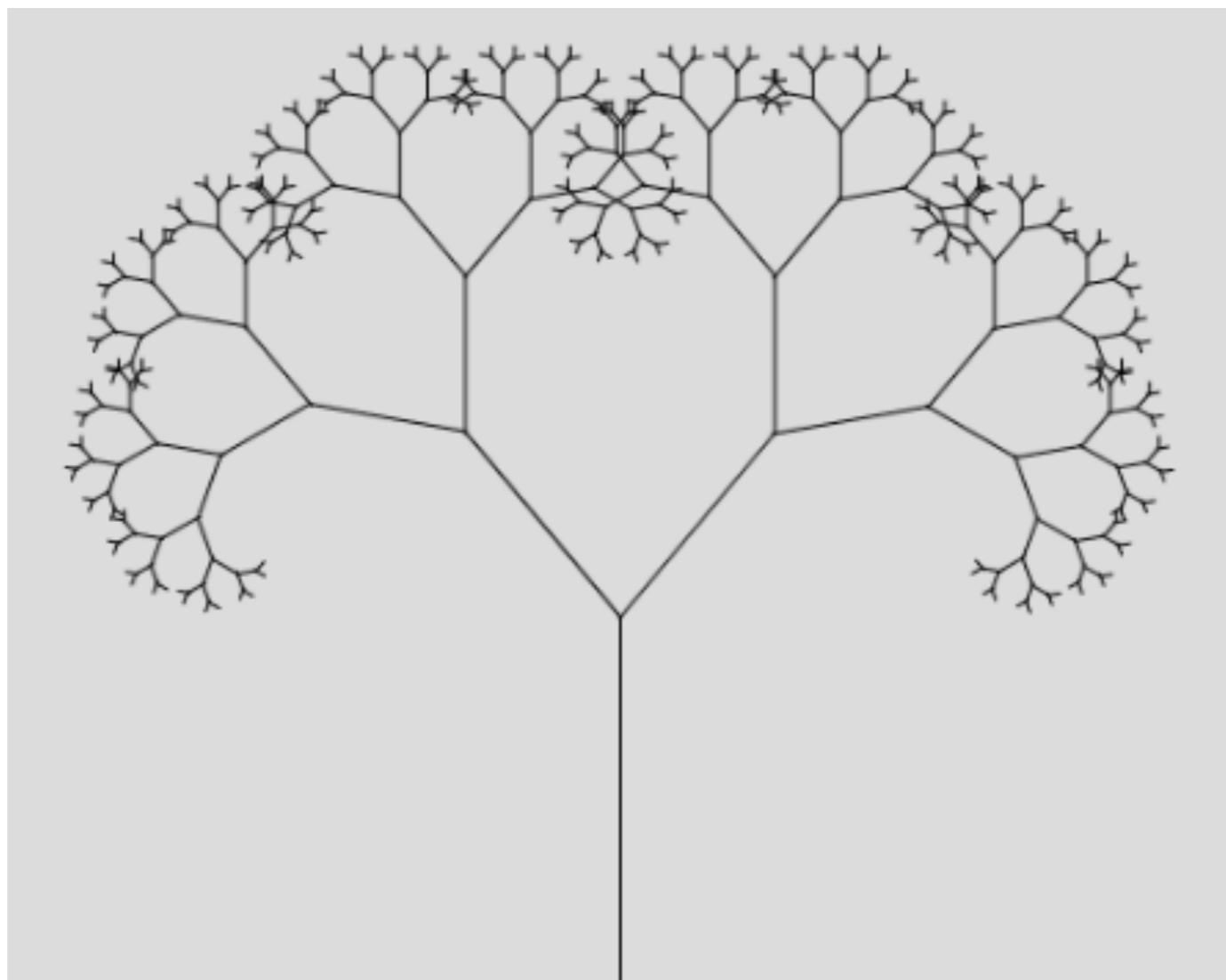
Number: upper bound of the value's target range

FRACTAL TREE



now lets use some random() variation
to make it look more natural...

FRACTAL TREE



now lets use some noise() to make it
give it some natural motion...

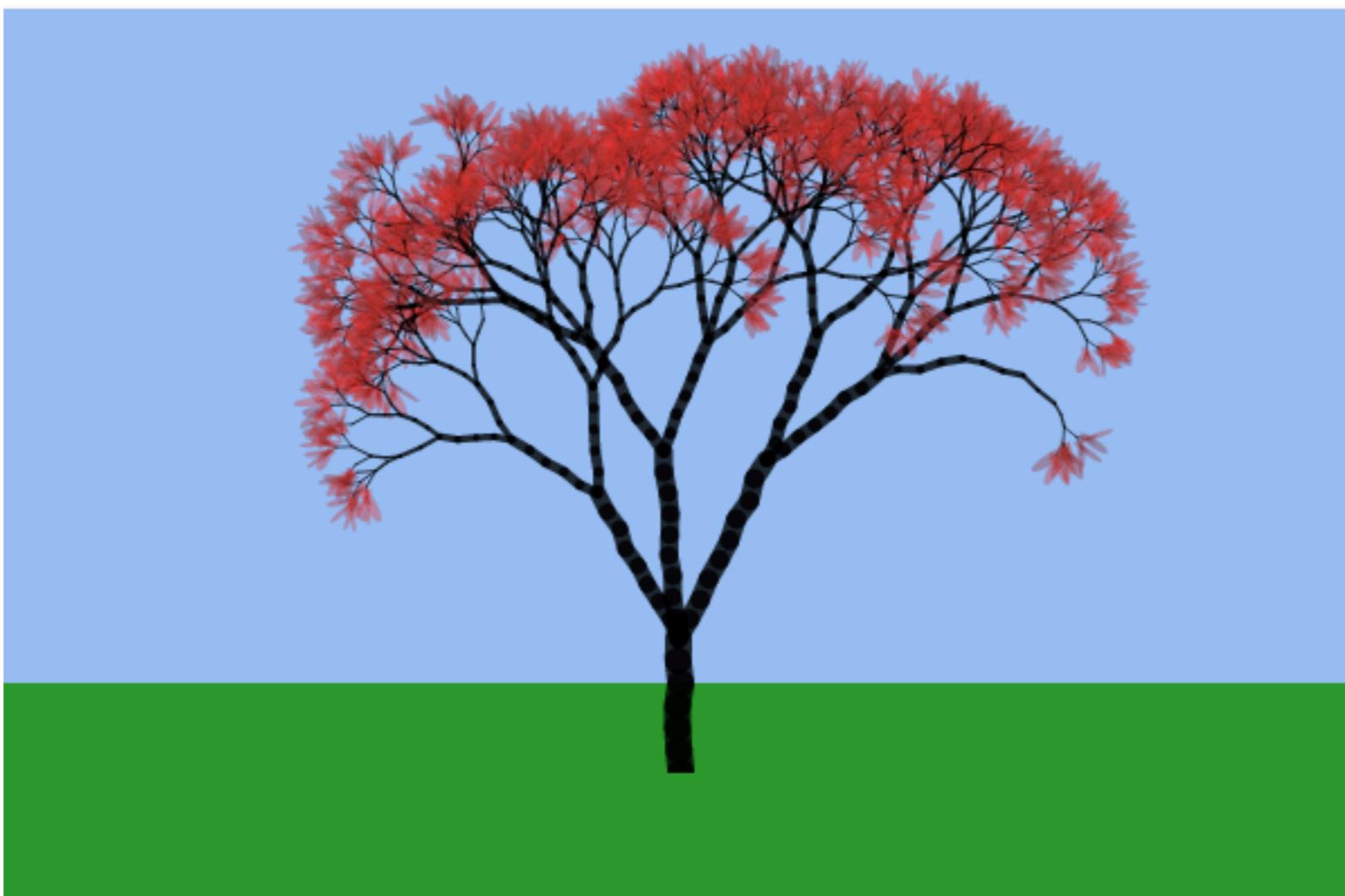
THE NOISE() FUNCTION

Description

Perlin noise is a random sequence generator producing a more natural ordered, harmonic succession of numbers compared to the standard random() function.

```
// 1d noise, 'value' should increase slowly  
let n = noise(value);
```

FRACTAL TREE



END

DANIEL C. HOWE

email: daniel@rednoise.org

web: <https://rednoise.org/daniel>
twitter/mastodon: @danielchowe