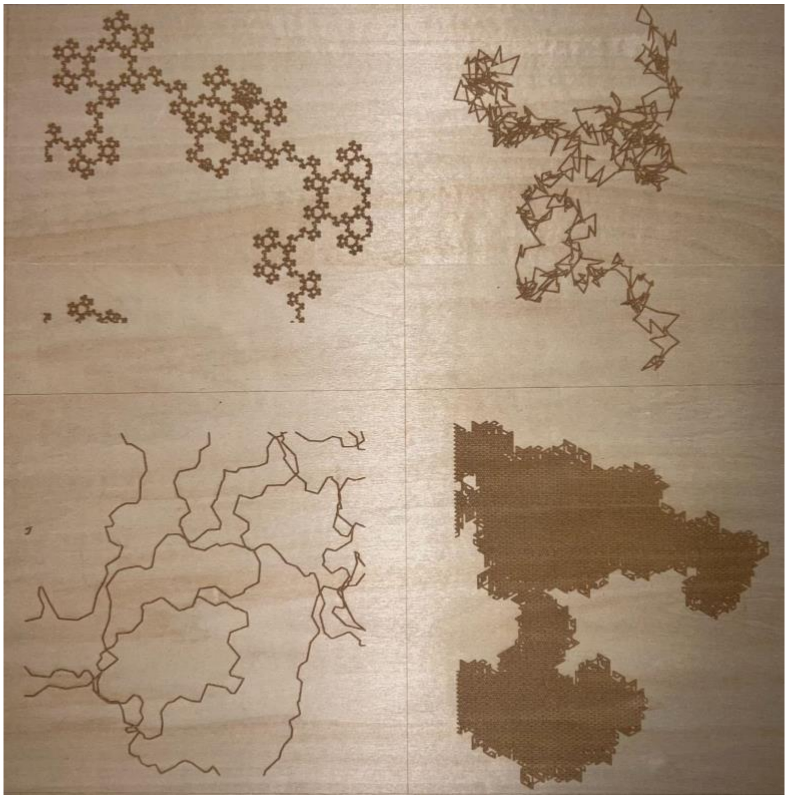
L1 — L-Systems [Link to gist]

L-Systems is the first large project of the semester, its purpose is to allow students to taste the power of computational design. Utilizing algorithms and turtle functions to generate iterative, branching patterns, designing L-Systems opens up a world of possibilities. Further, fabricating multiple designs provides yet another opportunity to use the laser cutters and to fabricate physical objects as a result of our computations.



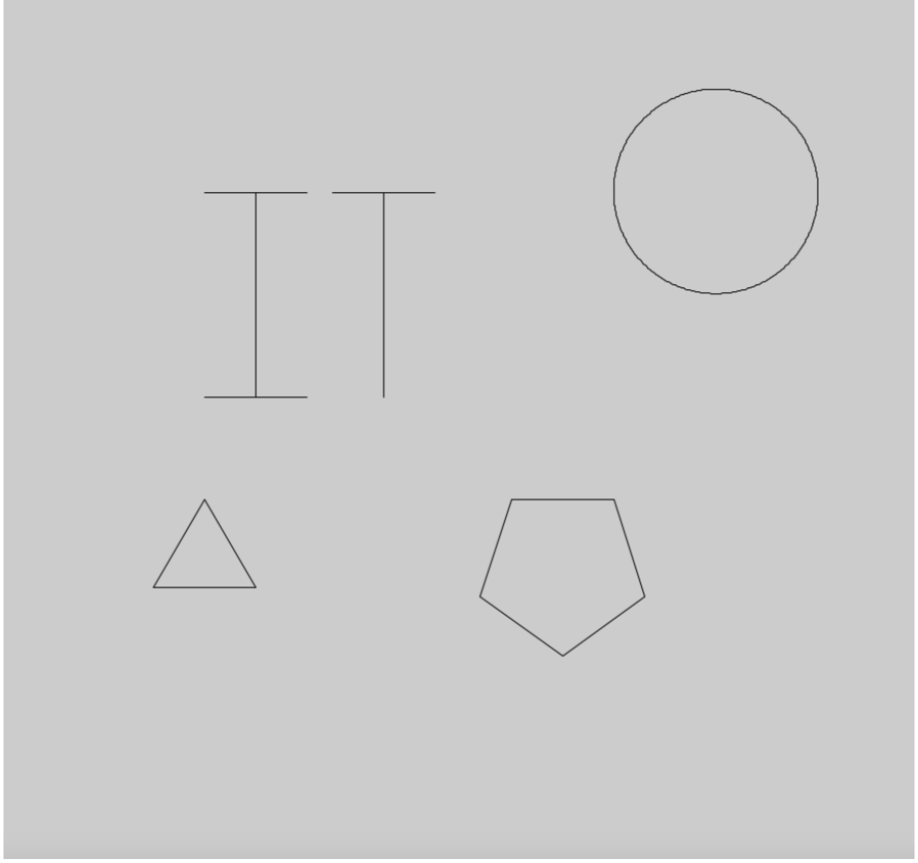
Materials Used

- Plywood (2.8mm thick)

Math Used (Task 1 Only)

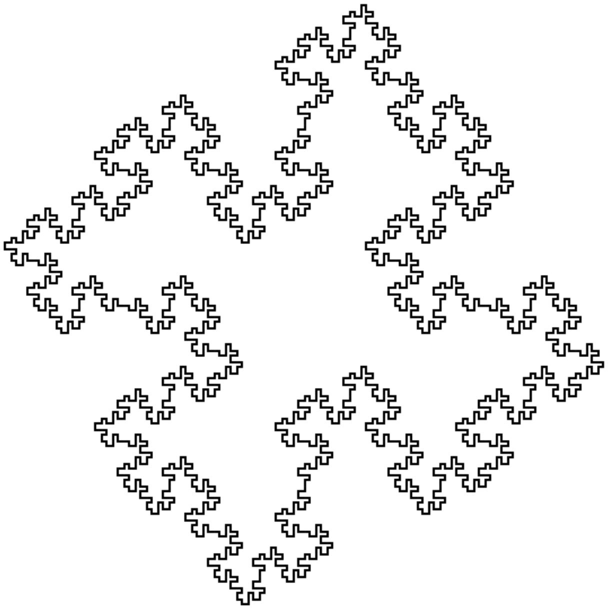
- Circle Circumference... $2\pi r$
- Internal Angle... $[(k-2)/k]*180$

Task #1



The first task was straightforward; write the word "IT," draw a triangle, a regular pentagon, and a circle with a defined radius. In doing so, we must not retrace steps, and should accomplish the task with penUp and penDown, and/or, push and pop. I chose to take advantage of penUp and penDown, coupled with the setX and setY turtle functions. As for the shapes, I used simple loops based on the shapes' internal angles, as well as the circumference equation for a circle to define a radius, allowing me to draw the circle based on it, rather than looping through small movements and angle changes.

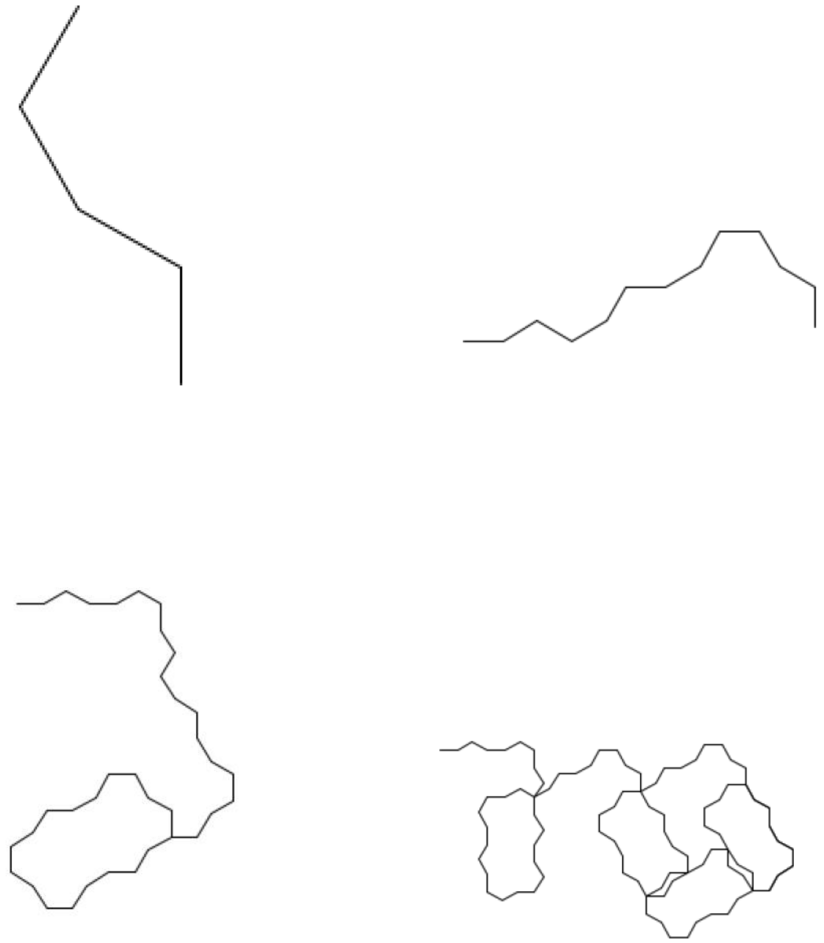
Task #2



Task two focused on understanding LSystems, both how they operate, as well as how to build them. With a given set of rules ready to be implemented, we followed to-dos which pointed us in the right direction, implementing the right vocabulary and creating initialization methods, to give a few examples. This exercise was quite pleasing once I had figured out the code, getting me excited to start playing around with my own LSystem designs in task three.

Task #3

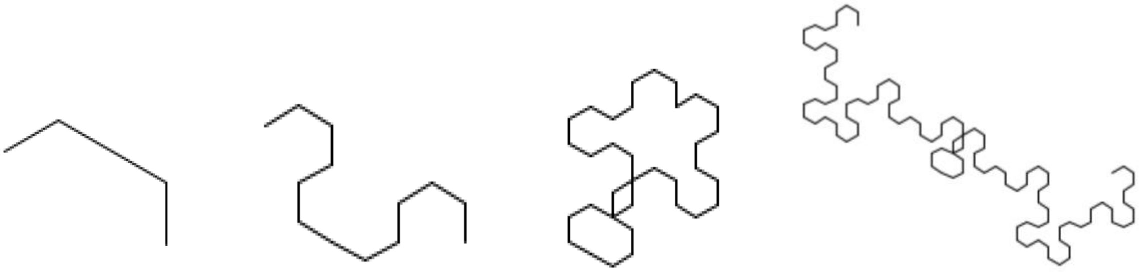
System 1: “LSystem first()”

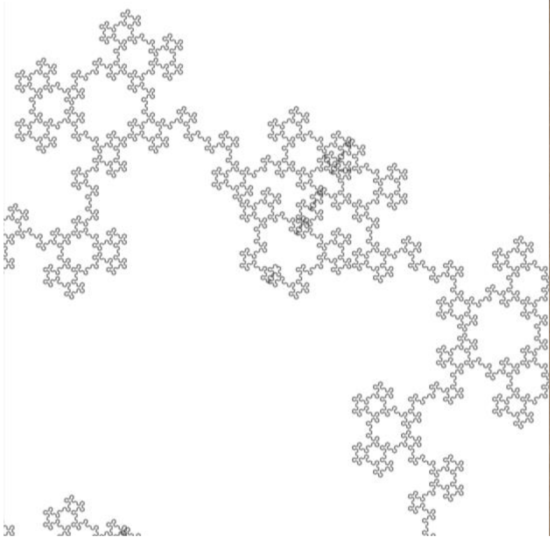


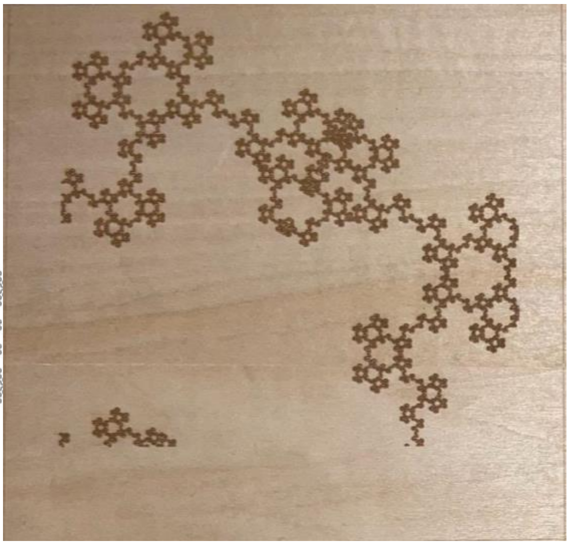
```
Drawing L-System: n =  0
n =  0  :  F--F+F++F
Drawing L-System: n =  1
n =  1  :  F--F+F--F--F+F+F--F+F++F--F+F
Drawing L-System: n =  2
n =  2  :  F--F+F--F--F+F+F--F+F--F--F+F--F--F+F+F--F+F+F--F+F--F--F+F+F--F+F++F--F+F--F--
F+F+F--F+F
Drawing L-System: n =  3
n =  3  :  F--F+F--F--F+F+F--F+F--F--F+F--F--F+F+F--F+F+F--F+F--F--F+F+F--F+F--F--F+F--F--
F+F+F--F+F--F--F+F--F--F+F+F--F+F+F--F+F--F--F+F+F--F+F+F--F+F--F--F+F+F--F+F--F--F+F--
F--F+F+F--F+F+F--F+F--F--F+F+F--F+F++F--F+F--F--F+F+F--F+F--F--F+F--F--F+F+F--F+F+F--
F+F--F--F+F+F--F+F
```

For my first L-System design, I went with a simple exploration of rules and iterations on the axiom which I think turned out quite well. It reminds me, almost, of the honeycomb-turtle example from class, although appearing like two, combined honeycombs at a time, wrapping themselves around each other, almost spiraling with each iteration. In creating this, I moved based on 30 degree angles, 50 step movements, and inputs of F, -, and +.

System 2:   “LSystem two()”



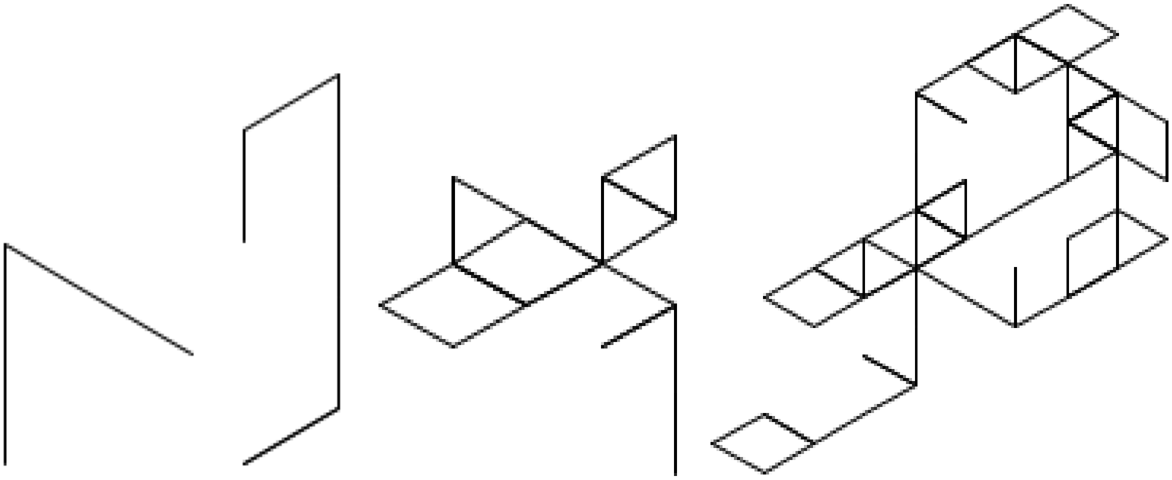


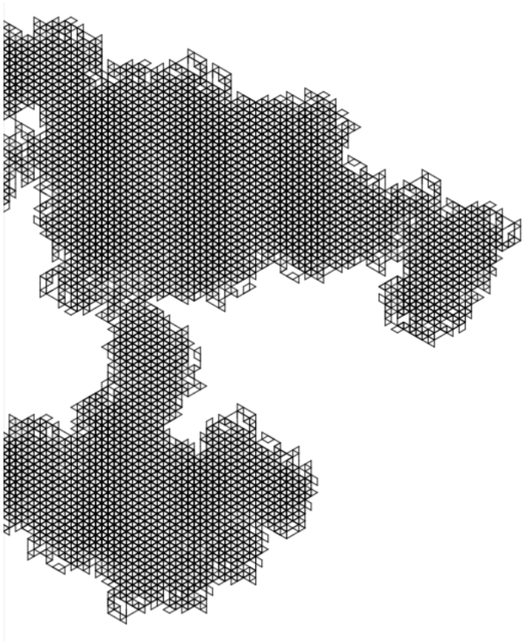


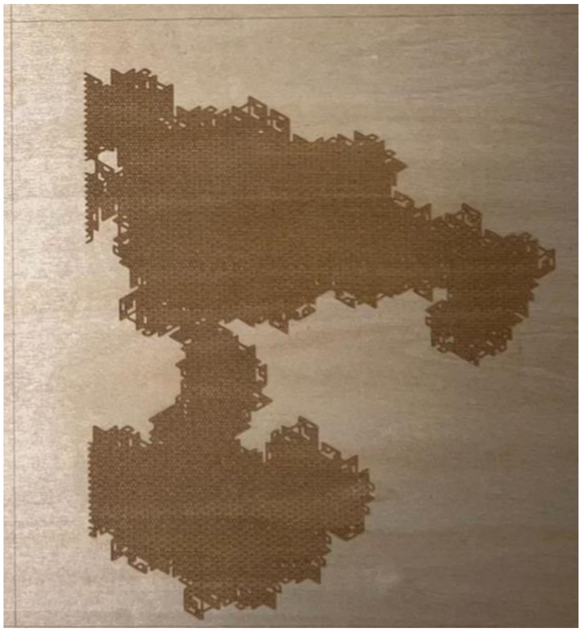
```
Drawing L-System: n =  0
n =  0  :  F-BB-F
Drawing L-System: n =  1
n =  1  :  F-B-F-B+F+BB+F+B-F-B-F
Drawing L-System: n =  2
n =  2  :  F-B-F-B+F+B-F-B-F-B+F+B+F-B-F+B+F+BB+F+B+F-B-F+B+F+B-F-B-F-B+F+B-F-B-F
Drawing L-System: n =  3
n =  3  :  F-B-F-B+F+B-F-B-F-B+F+B+F-B-F+B+F+B-F-B-F-B+F+B-F-B-F-B+F+B+F-B-
F+B+F+B+F-B-F-B+F+B-F-B-F+B+F+B+F-B-F+B+F+BB+F+B+F-B-F+B+F+B+F-B-F-B+F+B-F-
B-F+B+F+B+F-B-F+B+F+B-F-B-F-B+F+B-F-B-F-B+F+B+F-B-F+B+F+B-F-B-F-B+F+B-F-B-F
```

For my second L-System, I doubled my rotation angle, and added a fourth input character, “B”, while decreasing the distance covered on each move forward slightly. As expected, by doubling the first system's angle from 30 to 60, parts of hexagons were still being generated. However, with the angle increase and the addition of “B”, the movement was no longer consistent, creating an intricate and quite intriguing snowflake fractal as the iterations went on. I quite enjoyed fabricating this particular design, finding it to be extremely visually pleasing.

System 3:		“LSystem third()”



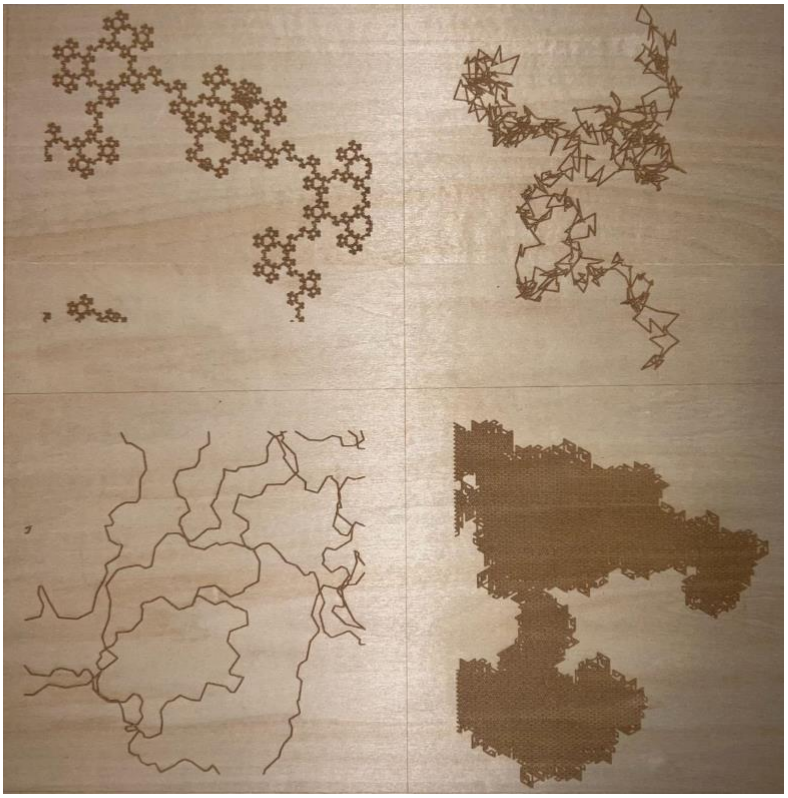




```
Drawing L-System: n =  0
n =  0  :  FCB
Drawing L-System: n =  1
n =  1  :  F+BCFFB+FC+BC
Drawing L-System: n =  2
n =  2  :  F+BCF++FC+BCFBF+BCFF+BCF+FC+BC+F+BCFFB++FC+BCFB
Drawing L-System: n =  3
n =  3  :
F+BCF++FC+BCFBF+BCF++F+BCFFB++FC+BCFBF+BCF+FC+BCF+BCF++FC+BCFBF+B
CFF+BCF++FC+BCFBF+BCF+F+BCFFB++FC+BCFB+F+BCF++FC+BCFBF+BCFF+BCF+F
C+BC++F+BCFFB++FC+BCFBF+BCF+FC+BC
```

For my third L-System, I decided to add a fifth input character, “C”, whose rule also included the addition of a doubled rotation angle. Inspired by the hexagons of its predecessors, by leaving the angle at 60 degrees, the doubled angle allowed for 120 degree turns and the creation of triangles; exactly what I was hoping for. As I did with the second L-System, I went much further than three iterations to create a more detailed pattern, making me similarly excited to fabricate this one. Although it was too visually dense for the size of plywood I was working with, the outcome succeeded in providing an extremely satisfying texture, if not accurately representing the L-System itself.

Task #4

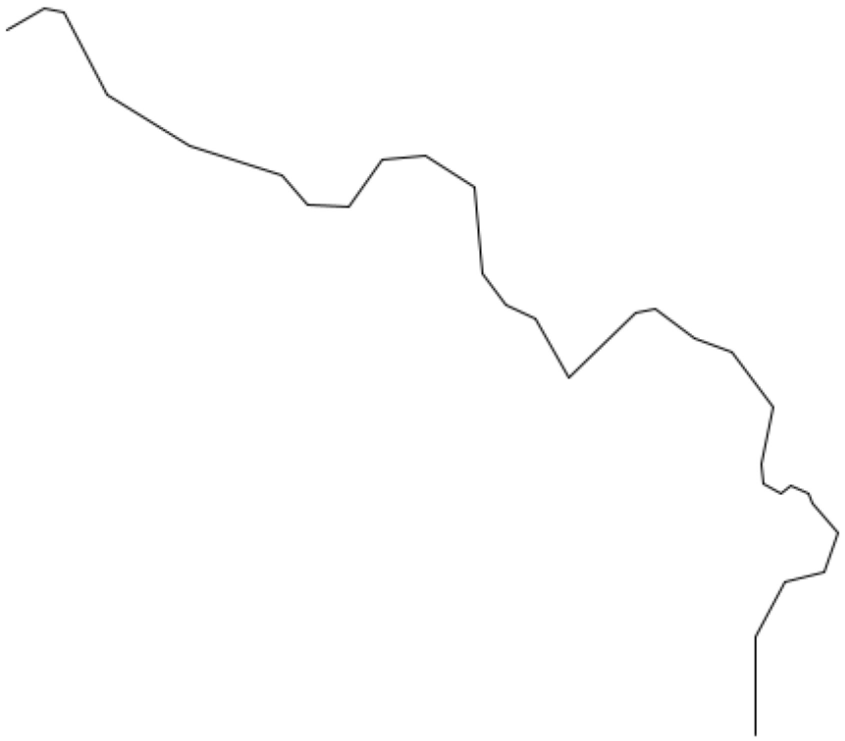


Seen above are my fabricated L-Systems. The top-left and bottom-right are LSystem two() and LSystem third(), respectively, and the other two are the result of my probabilistic L-Systems, probOne() in the bottom-left and probTwo() in the top-right. All were fabricated on a plywood sheet roughly 16 inches by 16 inches and 2.8 millimeters in thickness. I quite enjoy the laser cutting process, finding it to be satisfying, even mesmerizing at times, and this project once again provided quite an enjoyable experience fabricating.

Extra Credit: Probabilistic L-Systems

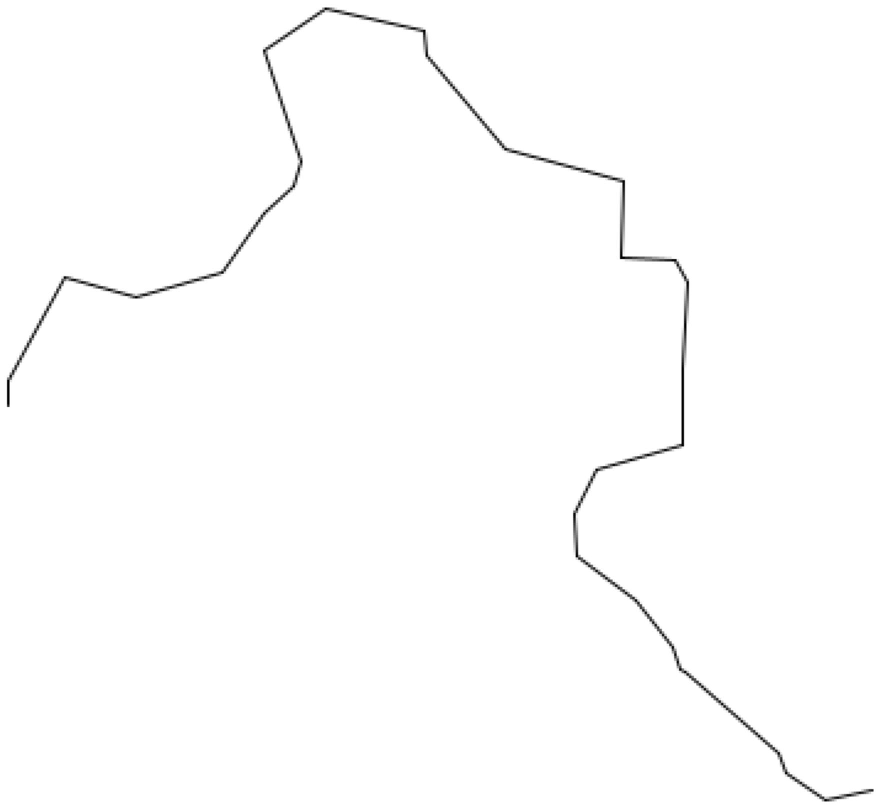
System 1: “ProbabalisticSystem probOne()”

Output #1



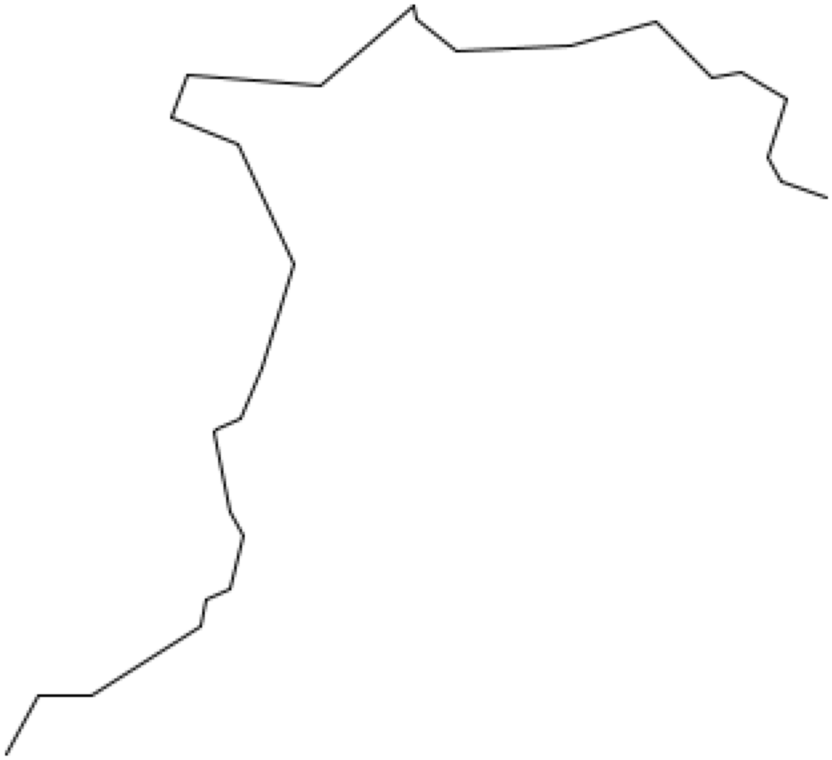
```
Drawing L-System: n =  0
n =  0  :  F+F+B-B
Drawing L-System: n =  1
n =  1  :  B-F+B-F+B+F-B+F
Drawing L-System: n =  2
n =  2  :  B+F-B-F+B+F-B-F+B+F+B-F-B+F+B-F
Drawing L-System: n =  3
n =  3  :  B+F+B-F-B+F-B-F+B+F+B-F-B+F-B-F+B+F+B-F+B+F-B-F-B+F+B-F+B+F-B-F
```

Output #2



```
Drawing L-System: n =  0
n =  0  :  F+F+B-B
Drawing L-System: n =  1
n =  1  :  B-F+B-F+B+F-B+F
Drawing L-System: n =  2
n =  2  :  B+F-B-F+B+F-B-F+B+F+B-F-B+F+B-F
Drawing L-System: n =  3
n =  3  :  B+F+B-F-B+F-B-F+B+F+B-F-B+F-B-F+B+F+B-F+B+F-B-F-B+F+B-F+B+F-B-F
```

Output #3

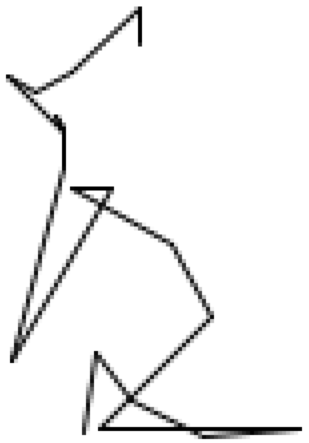


```
Drawing L-System: n =  0
n =  0  :  F+F+B-B
Drawing L-System: n =  1
n =  1  :  B-F+B-F+B+F-B+F
Drawing L-System: n =  2
n =  2  :  B+F-B-F+B+F-B-F+B+F+B-F-B+F+B-F
Drawing L-System: n =  3
n =  3  :  B+F+B-F-B+F-B-F+B+F+B-F-B+F-B-F+B+F+B-F+B+F-B-F-B+F+B-F+B+F-B-F
```

For my first probabilistic L-System, I focused the randomness to within the instruction for the turtle operations, adding random number generation,—some coupled with mathematical operations—and by pulling from arrays. My goal with the probabilistic systems was to make the generation as random as possible, and I think I did a decent job getting off on the right foot with this one. Although I don't find this system particularly pretty, occasionally, like the one used for fabrication, they appear like beautiful cascades of lightning strikes.

System 2: “ProbabalisticSystem probTwo()”

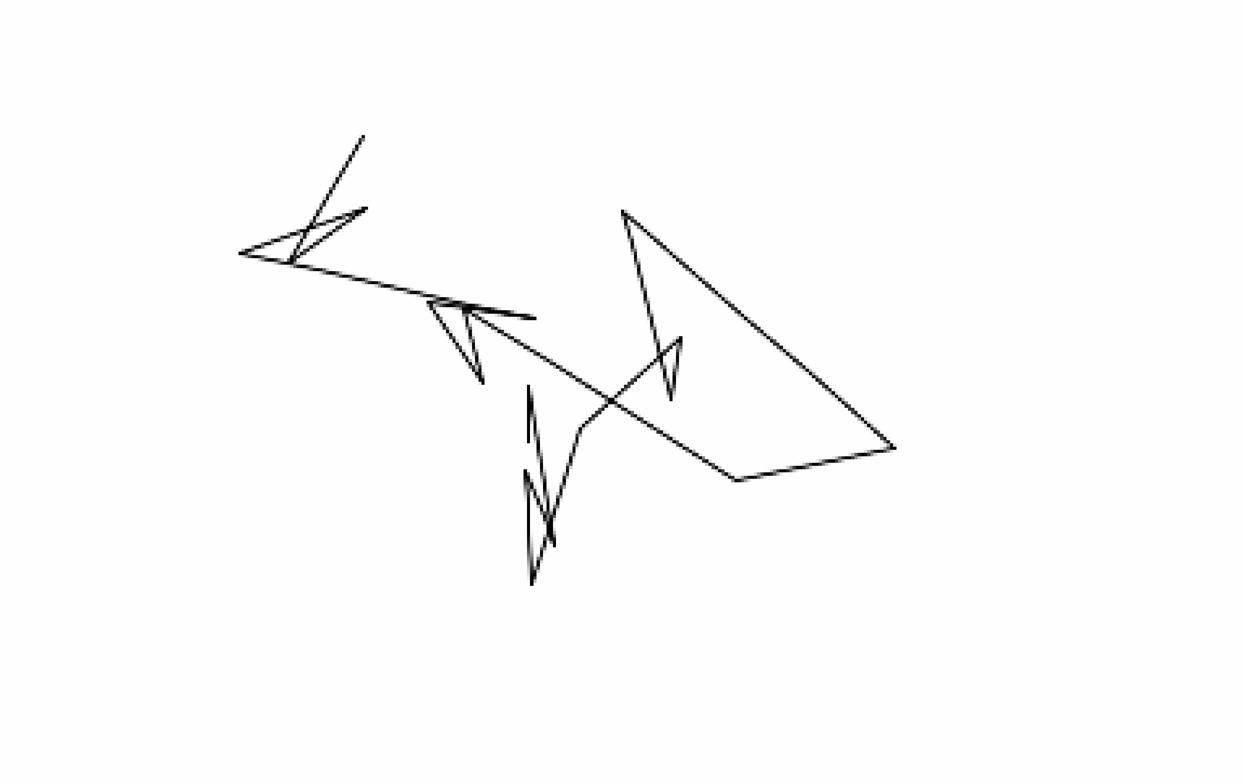
Output #1



```
Drawing L-System: n =  0
n =  0  :  F+F+B-B
Drawing L-System: n =  1
n =  1  :  F+C+F+C+B+C-B+C
Drawing L-System: n =  2
n =  2  :  F+C+F+B+F+C+F+B+B+C+F+B-B+C+F+B
Drawing L-System: n =  3
```

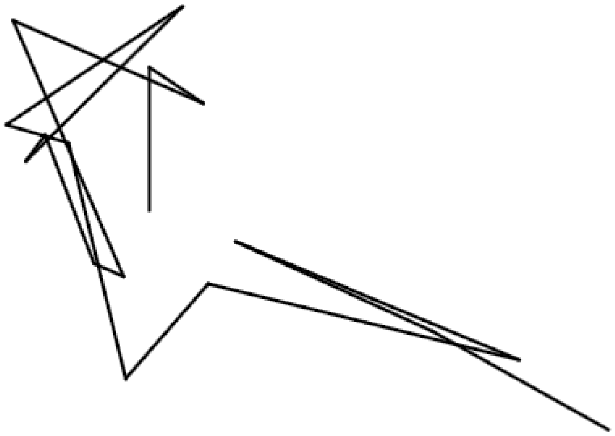
```
n =  3  :  F+C+F+B+F+C+B+C+F+C+F+B+F+C+B+C+B+C+F+B+F+C+B+C-B+C+F+B+F+C+B+C
```

Output #2



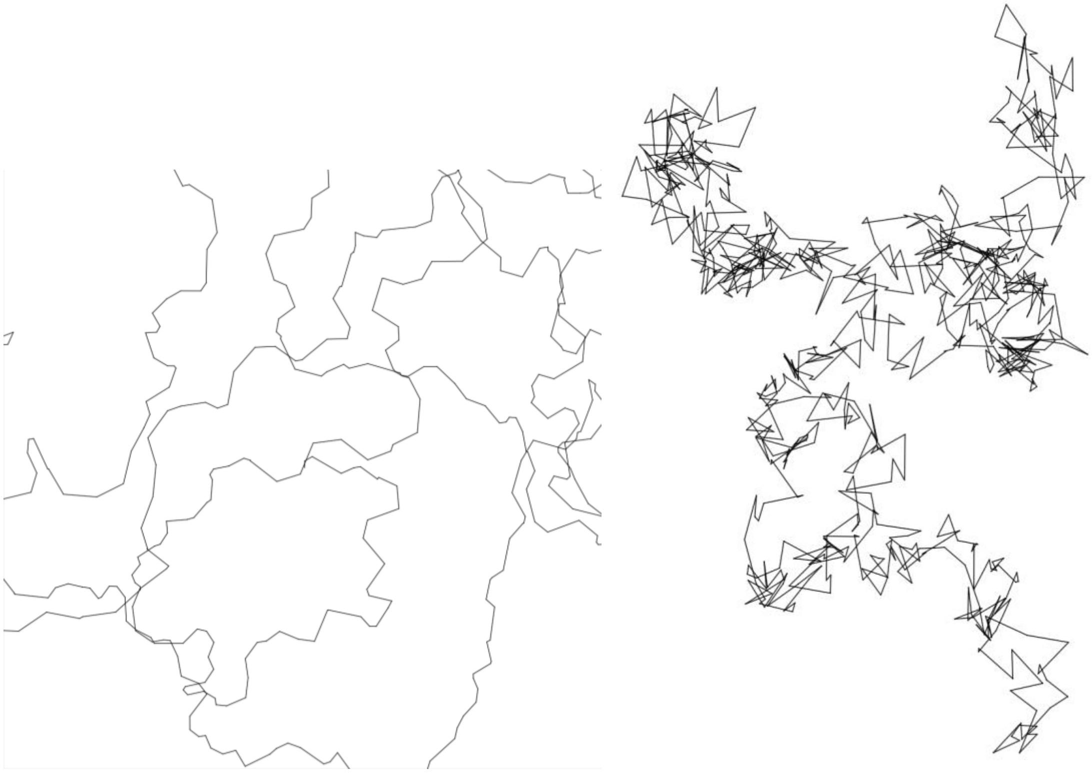
```
Drawing L-System: n =  0
n =  0  :  F+F+B-B
Drawing L-System: n =  1
n =  1  :  F-B+F-B+BF-BF
Drawing L-System: n =  2
n =  2  :  F-B-BF+F-B-BF+BFF-B-BFF-B
Drawing L-System: n =  3
n =  3  :  F-B-BF-BFF-B+F-B-BF-BFF-B+BFF-BF-B-BF-BFF-BF-B-BF
```

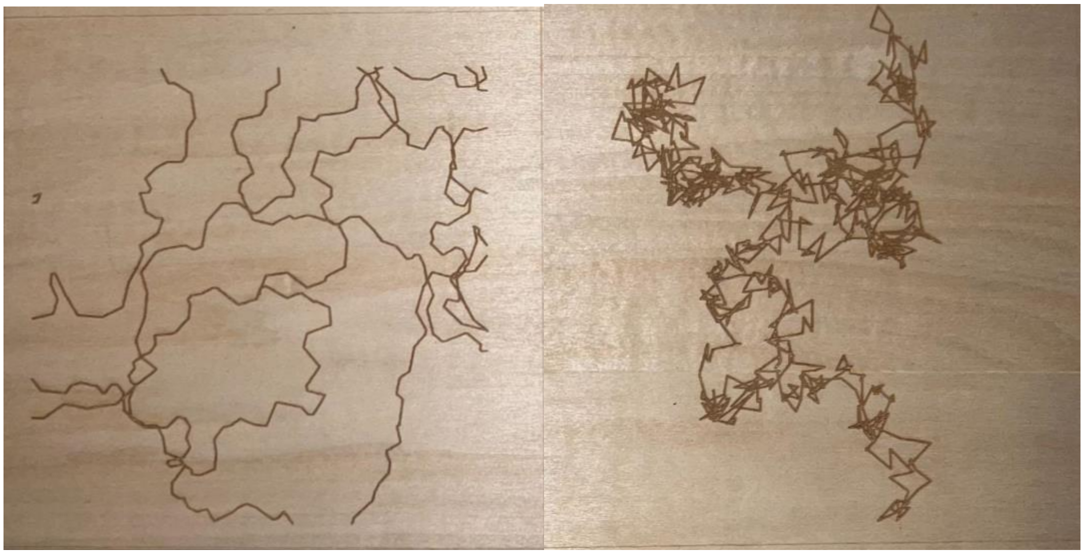
Output #3



```
Drawing L-System: n =  0
n =  0  :  F+F+B-B
Drawing L-System: n =  1
n =  1  :  F-B+F-B+BF-BF
Drawing L-System: n =  2
n =  2  :  F-B-BF+F-B-BF+BFF-B-BFF-B
Drawing L-System: n =  3
n =  3  :  F-B-BF-BFF-B+F-B-BF-BFF-B+BFF-BF-B-BF-BFF-BF-B-BF
```

For my second probabilistic L-System, I decided to include randomization within the rules of the string iteration definitions. By declaring a float p randomized each run of the L-System between 0.0 and 1.0, as well as by declaring an array of string choices, I was able to randomize the generation of both the string and the operations, allowing for entirely different L-Systems each time it is run. I found the folding-in-on-itself of the lines to be incredibly interesting and intricate, reminding me of Picasso's line work.





Problems/Conclusion

In all honesty, I did not face too many problems while tackling this first large assignment. The initial to-do of implementing the replacement procedure took me little while to think through and to find the right members to call, but once I had worked it out, it was a pretty straightforward process. One other thing that caught me off guard was creating a circle with a defined radius, and what that meant exactly. After a bit of brainstorming with a few classmates, it similarly became clear, and I was able to continue on. All in all, I definitely enjoyed working with L-Systems, and would like to continue exploring their potential in the future.