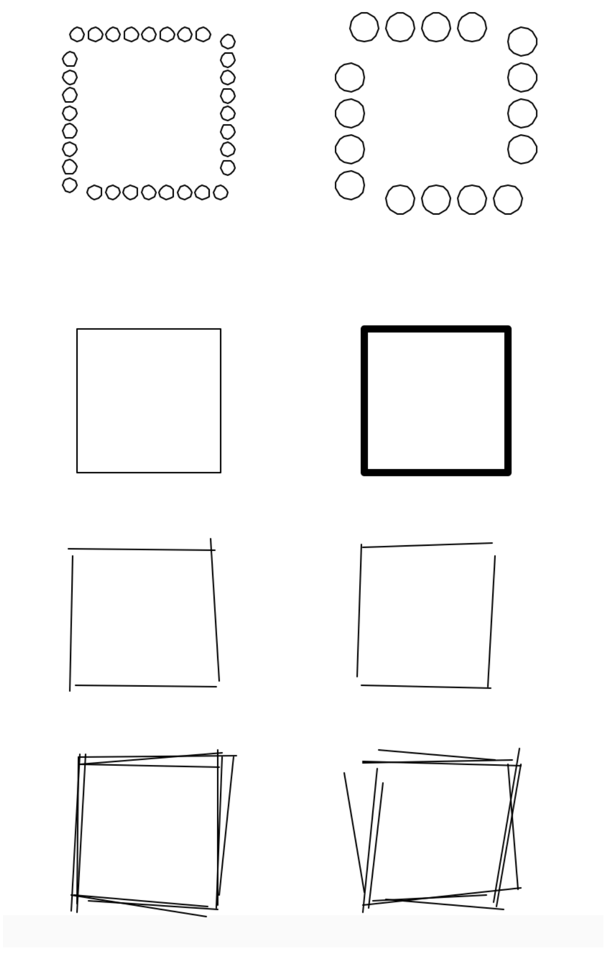
Natalie's CS151 Project 10

Title and Abstract: Project 10- non-photorealistic rendering. For this project, we coded two new styles of drawing for the turtle and enhanced our scene from project 9.

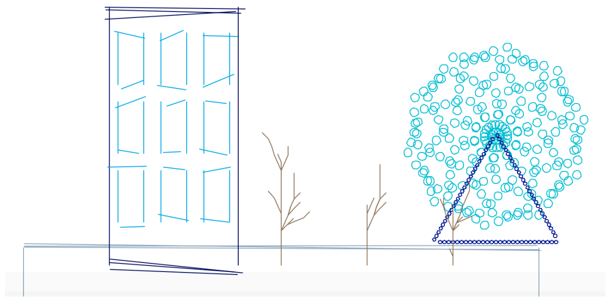
For the first few tasks, I created a 'jitter3' and 'dotted' style. The 'jitter3' style draws three lines at a random distance (with bounds set by the jitterSigma value) from where the endpoints of the shape would normally be with the 'normal' style. The 'dotted' style draws the lines instead as multiple dots, within this style, there is a setDotSize method which sets the dot size and then calculates how many dots can fit on said line length with the size of each dot. For example, the bigger the dot size, the less dots it will take to draw each line. I included a code snip bit:

```python
elif self.style == 'dotted':
    x0,y0 = turtle.position()
    numDots = int(distance/(self.dotSize*2.5))
    for i in range(numDots):
        turtle.pendown()
        turtle.circle(self.dotSize)
        turtle.penup()
        turtle.forward(self.dotSize*2.5)
```

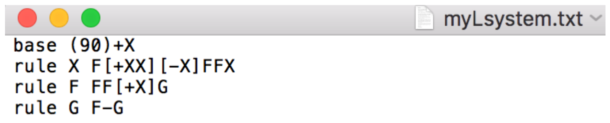
My image below demonstrates two of each style drawn as squares. The first one is using the 'dotted' style with dotSize = 5 and 10. The second set is a 'normal' style with lineWidth= 1 and 5. The second to last set uses 'jitter' style from with jitter= 5 and 7. Finally the last set of squares has a 'jitter3' setting with jitter' 5 and 7 as well. These styles are included in this project because by making a jitter or a bunch of small dots, the scene can now look more as if someone has sketched it rather than a program.



For task four, I enhanced my scene from Project 9 by applying my filters to the shapes in it. I decided to take out the fill aspect from project 9 because I wanted my viewer to be able to see the jitter more fully. I chose to use 'jitter3' on the road and building and just 'jitter' on the windows. I chose these styles because I feel the 'jitter3' gives a harsher line than the 'jitter' since it has 3 lines compared to 1. I then chose to apply the 'dotted' style to the ferris wheel. The Navy Pier ferris wheel alwyas has a fun light pattern on it, and I thought the dots were a good representation of this. This again uses what we learned in lecture because by updating the turtle_interpreter, I was able to write a simple code to call upon these styles in a different file.

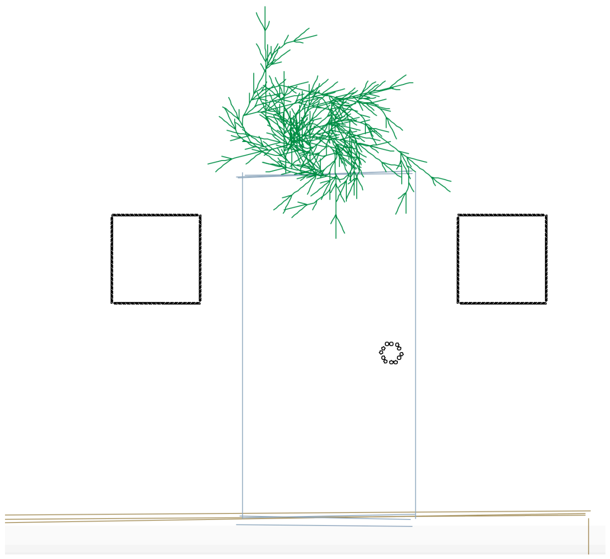


For task five, I created a new Lsystem to draw a wreath. To create do this, I made a variation of a system from the ABOP book. I did a bit of playing around with the system before settling on this:



```
base (90)+X
rule X F[+XX][-X]FFX
rule F FF[+X]G
rule G F-G
```

I then made a scene utilizing the various shapes and styles from above and placed my new lsystem at the top of the door like a wreath. I again used the 'jitter' and 'dotted' styles to give a sketch like appearance to the scene. This was my first time manipulating an lsystem with multiple rules.



What I learned: I learned how to apply different styles to previous code without completely changing it's structure. These style changes were an add on to the turtle_interpreter and therefore I was able to call upon them to modify existing scenes. I also learned how to manipulate lsystems to get them to turn, break, and extend how I want them to. I learned this while manipulating my lsystem in task 5.

Collaborators: Andrew Turley, Emmett Burnes, Zeb Keith-Hardy. They all helped me with bugs and Andrew specifically with the 'dotted' style in task one. He helped explain to me how and why to call upon the dotSize method before drawing to determine how many dots will be in each line.

I imported sys, shapes, turtle_interpreter, and lsystem to complete this project.