Lab Report

Diego Quinones

80582918

Recursive Plots.

Lab #1 consisted of plotting different shapes and certain patterns in the, these patterns consisted on recursive drawings, which where shapes inside shapes following a change in the values. Ex. circles inside circles, squares inside squares

Because I am not that familiar with code structure for plot, I started with playing with the values on the previously provided square and circle codes, trying to figure out how every variable affected the actual shape. Now that I got the hang of it I saw the instructions and tried to find a pattern on how shapes change, that way I can implement it to the recursive call, for example the second circles problem, show each new circle being one third of the size of previous one and, and moving on third back, so I made those changes on the recursive call, and they would loop thanks to a variable called n. n was implemented since the first method call and is inputed prior.

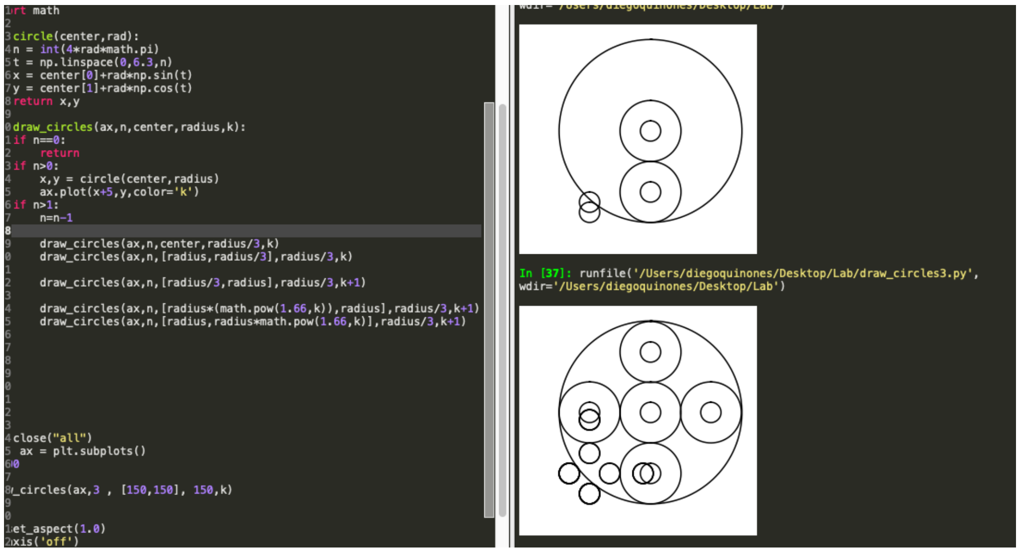
Most of the trouble I had was thanks to not being familiar with how plots work, and also square and circle plot codes being noticeable different, made harder to translate what is supposed to be the same type of logic on these different problems. Also another issue I faced was not noticing patterns on first hand. On the 5 circles problem it took me so much time, because I thought the size of circles was only being divided but it was being divided and then subtracted.

On this first lab I learned to use plots and to write recursive formulas. What I also learned that I need to practice more.

Running Times Respectively

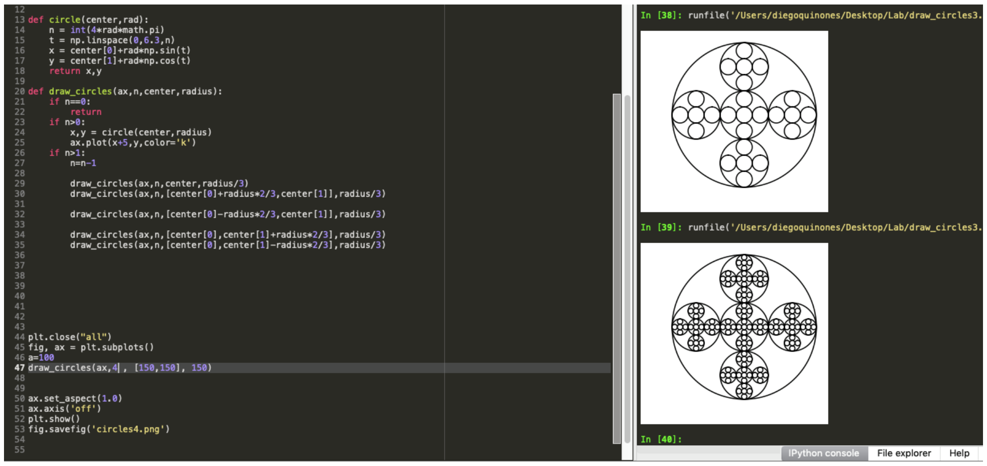
0.25308799743652344 seconds
0.39146995544433594 seconds
0.7099668979644775 seconds
0.9884839057922363 seconds
1.6557419300079346 seconds
2.1305081844329834 seconds
2.265976905822754 seconds
2.70241117477417 seconds
3.3072099685668945 seconds
3.442840814590454 seconds
3.7089669704437256 seconds
4.743278980255127 seconds
8.883681058883667 seconds
9.067010879516602 seconds

Tests



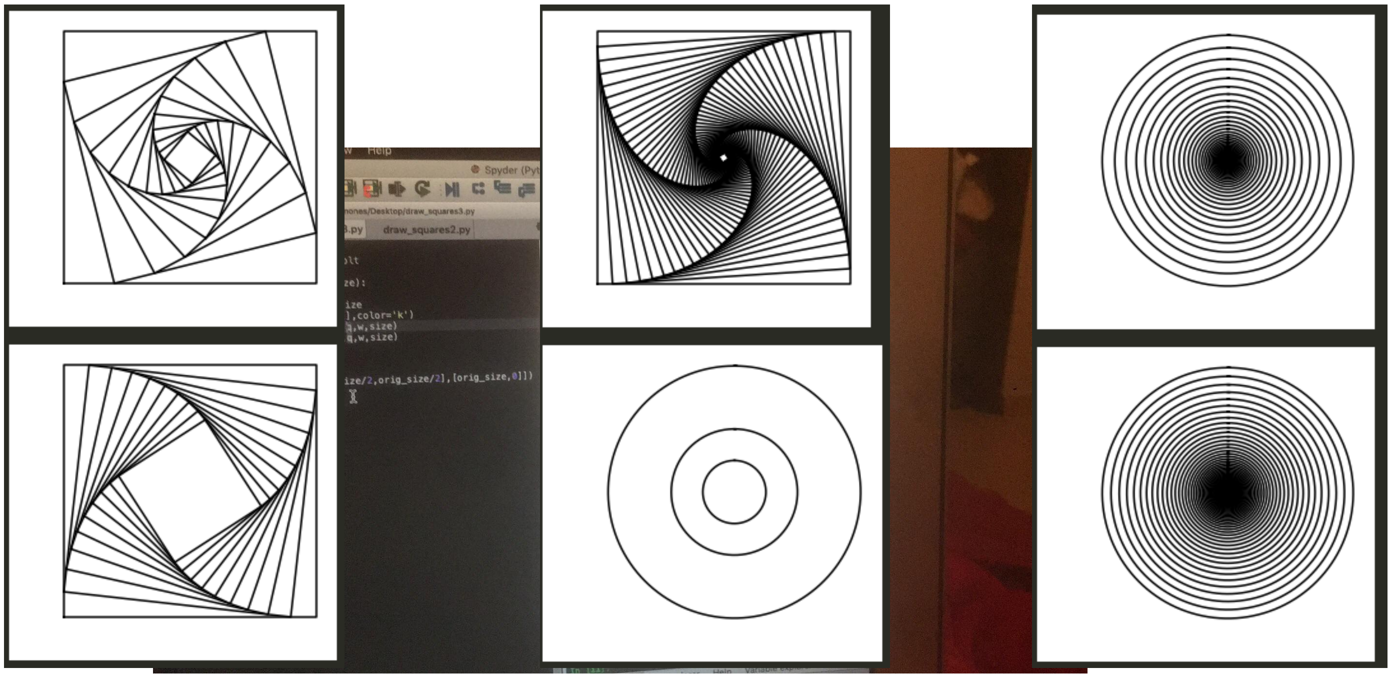
```
rt math

circle(center,rad):
n = int(4*rad*math.pi)
t = np.linspace(0,6.3,n)
x = center[0]+rad*np.sin(t)
y = center[1]+rad*np.cos(t)
return x,y

draw_circles(ax,n,center,radius,k):
if n==0:
    return
if n>0:
    x,y = circle(center,radius)
    ax.plot(x+5,y,color='k')
if n>1:
    n=n-1

    draw_circles(ax,n,center,radius/3,k)
    draw_circles(ax,n,[radius,radius/3],radius/3,k)

    draw_circles(ax,n,[radius/3,radius],radius/3,k+1)

    draw_circles(ax,n,[radius*(math.pow(1.66,k)),radius],radius/3,k+1)
    draw_circles(ax,n,[radius,radius*math.pow(1.66,k)],radius/3,k+1)








close("all")
 ax = plt.subplots()
0

_circles(ax,3 , [150,150], 150,k)


et_aspect(1.0)
xis('off')
```

```
wdir='/Users/diegoquinones/Desktop/Lab')

In [37]: runfile('/Users/diegoquinones/Desktop/Lab/draw_circles3.py',
wdir='/Users/diegoquinones/Desktop/Lab')
```



```

def circle(center,rad):
    n = int(4*rad*math.pi)
    t = np.linspace(0,6.3,n)
    x = center[0]+rad*np.sin(t)
    y = center[1]+rad*np.cos(t)
    return x,y

def draw_circles(ax,n,center,radius):
    if n==0:
        return
    if n>0:
        x,y = circle(center,radius)
        ax.plot(x+5,y,color='k')
    if n>1:
        n=n-1

        draw_circles(ax,n,center,radius/3)
        draw_circles(ax,n,[center[0]+radius*2/3,center[1]],radius/3)

        draw_circles(ax,n,[center[0]-radius*2/3,center[1]],radius/3)

        draw_circles(ax,n,[center[0],center[1]+radius*2/3],radius/3)
        draw_circles(ax,n,[center[0],center[1]-radius*2/3],radius/3)








plt.close("all")
fig, ax = plt.subplots()
a=100
draw_circles(ax,4 , [150,150], 150)


ax.set_aspect(1.0)
ax.axis('off')
plt.show()
fig.savefig('circles4.png')


```

```
In [38]: runfile('/Users/diegoquinones/Desktop/Lab/draw_circles3.

In [39]: runfile('/Users/diegoquinones/Desktop/Lab/draw_circles3.

In [40]:
```



```
olt

ze):

ize
],color='k')
q,w,size)
q,w,size)



ize/2,orig_size/2],[orig_size,0]])
```