**Course Name:** 2302 **Author:** Olugbenga Iyiola **ID:** 80638542 **Instructor:** Olac Fuentes **TA:** Nath Anindita **LAB #1 Report**

**Introduction**

The purpose of the lab is to write various recursive programs to draw different fascinating shapes such as concentric circles, multiple squares etc. The program is written with python and uses packages such as Matplotlib, Numpy and Math. The program helps to understand how recursion works using appropriate base cases and making recursive calls to get the desired figures.

**Proposed Solution Design and Implementation**

Four different py files are used in this lab to get the desired figures and the algorithms and clearly separated to ensure clarity

The first figure was plotted using this algorithm;

* Define method using coordinates of circle center, width, and number of loop times as parameters.
* if n>0: # Method executes for n number of times & base

case

* draw\_squares (ax,center,width) # Plotting the square using the desired center

plot using x and y coordinates coordinates and width as parameter for method

call

* Create center coordinates for square’s left top corner, right top corner, right bottom corner and left bottom corner.
* Recursively call method with the four different centers for n times.

The second figure was plotted using the following algorithm;

* Define method using coordinates of circle center, radius, and the reducing factors as parameters
* if n > 0: # Method executes for n times  
   x , y = circle(center, radius) # Calling circle method with center and radius

parameters to create 2πRadius circumference  
 ax.plot(x, y, color='k') # Plotting the circle by joining coordinates with line  
 center = center - (radius\*(1-w)) # Shifting center of circle along x-axis inwardly.

* Recursively calling the method with new centers

The third figure was plotted using the following algorithm;

* Define method using coordinates of circle center, radius, and number of loop times as parameters
* if n>0: # Method executes n times  
   draw\_lines(ax,center,width) # Calling method to draw tree desired center and

width as parameters.

* Construct center coordinates for left and right children of tree.
* Recursively call the method with right and left center coordinates.

The fourth figure was plotted using the following algorithm;

* Define method using coordinates of circle center, radius, and number of loop times as parameters.
* if n>0: # Method executes for n number of times & base

case

* circle (center, radius) # Calling circle method with center and radius

parameters to create 2πRadius circumference  
 ax.plot(x, y, color='k') # Plotting the circle by joining coordinates with line

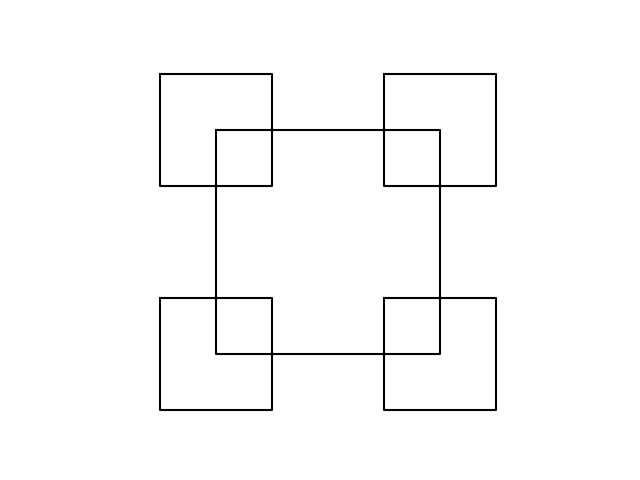
* Create center coordinates for circles at left corner, right corner, top and bottom corners.

* Recursively call method with the original center and four different new centers for n times.

**Experimental Result**

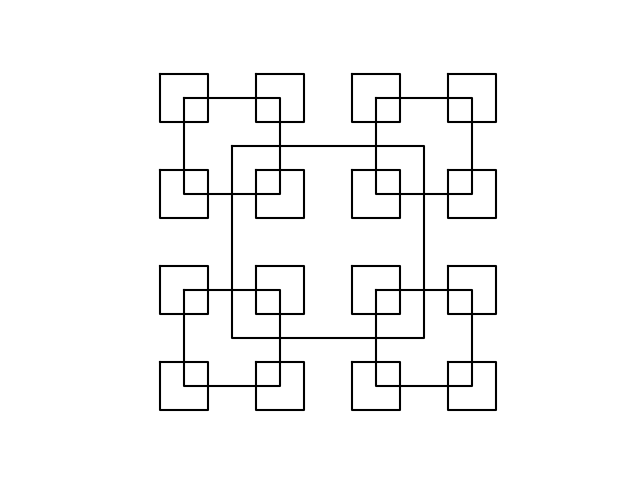
System Specification: HP Windows 10, 1.60GHZ Intel® Celeron® , 4.GB RAM, 64-bit operating system

The outputs for the programs are as shown below;



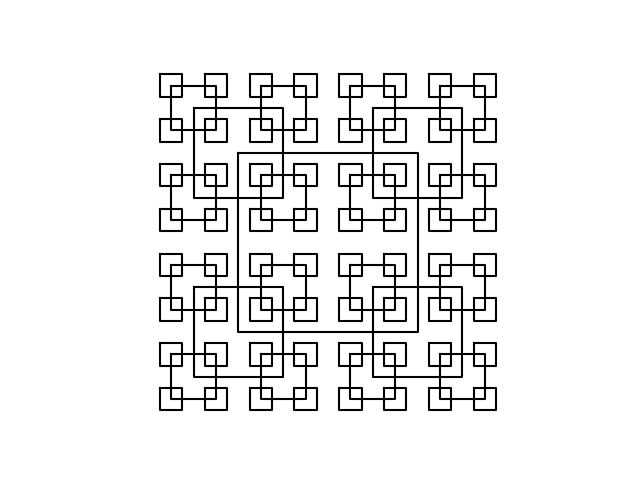
**T(n) = 4T(n – 1) + 1**

**Number of runtimes(n) = 2**



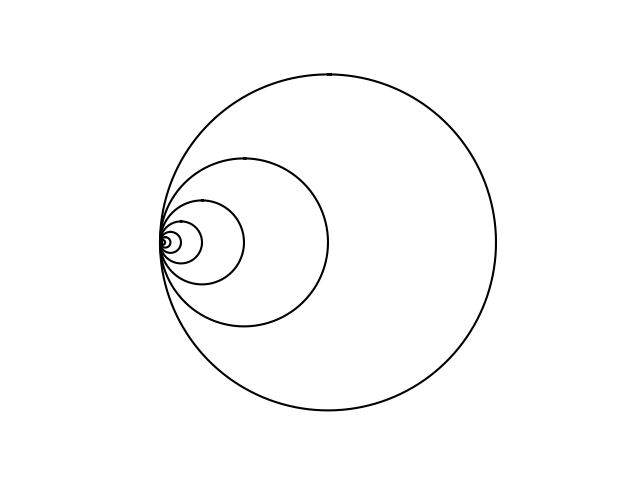
**T(n) = 4T(n – 1) + 1**

**Number of runtimes(n) = 3**



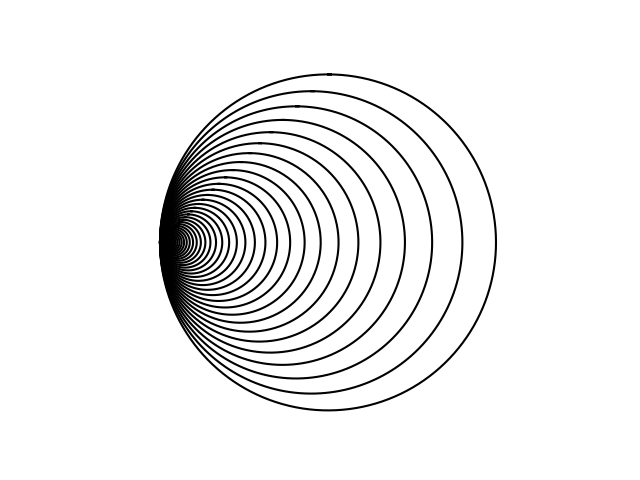
**T(n) = 4T(n – 1) + 1**

**Number of runtimes(n) = 4**



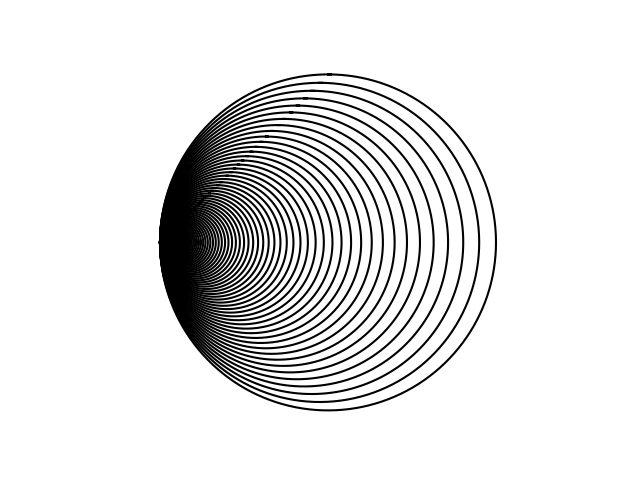
**T(n) = T(n – 1) + 1**

**Number of runtimes(n) = 200**



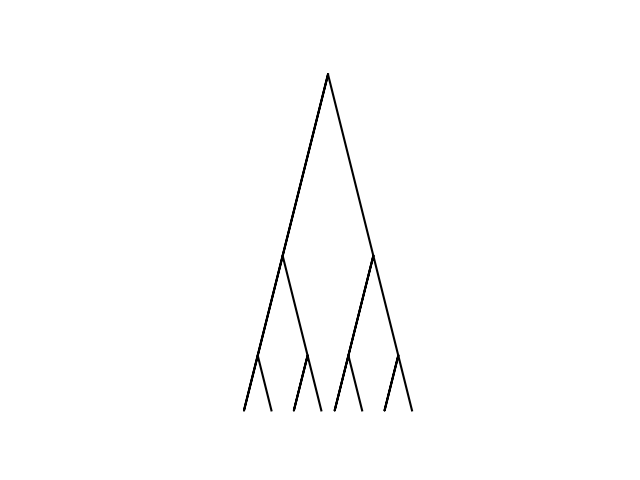
**T(n) = T(n – 1) + 1**

**Number of runtimes(n) = 200**



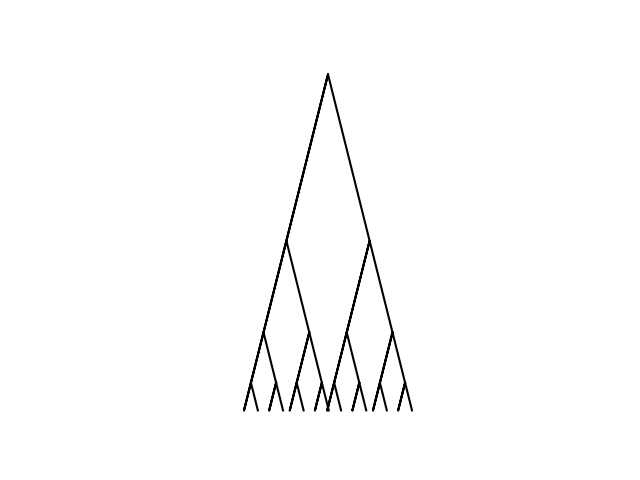
**T(n) = T(n – 1) + 1**

**Number of runtimes(n) = 200**



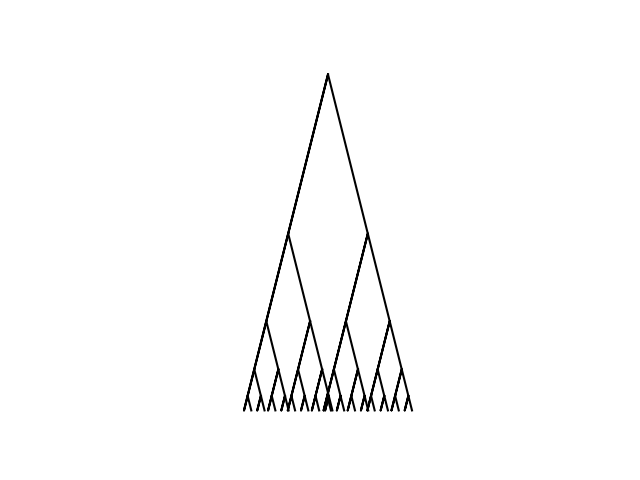
**T(n) = 2T(n – 1) + 1**

**Number of runtimes(n) = 3**



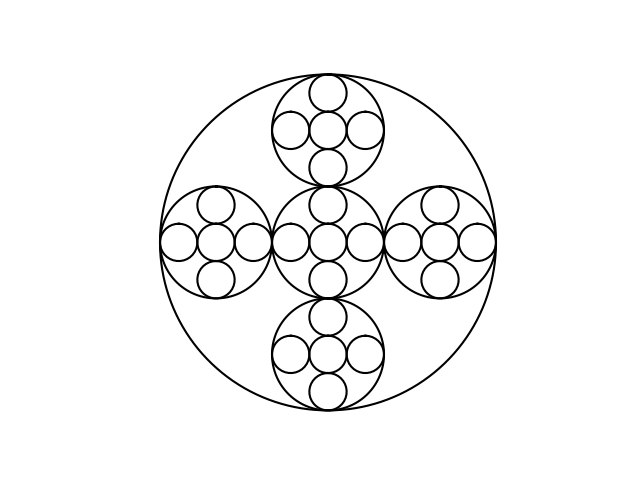
**T(n) = 2T(n – 1) + 1**

**Number of runtimes(n) = 4**



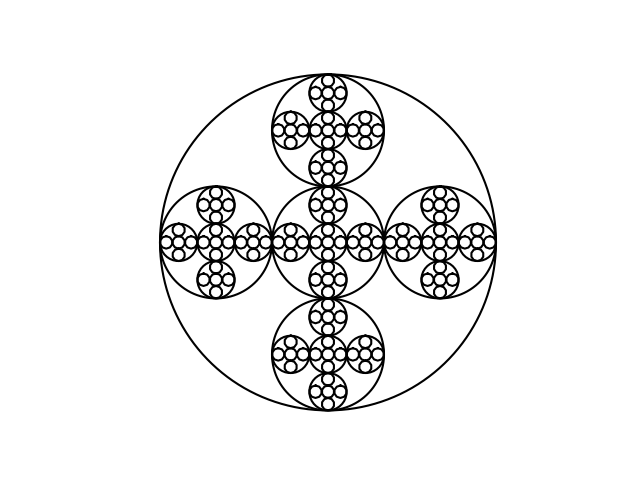
**T(n) = 2T(n – 1) + 1**

**Number of runtimes(n) = 7**



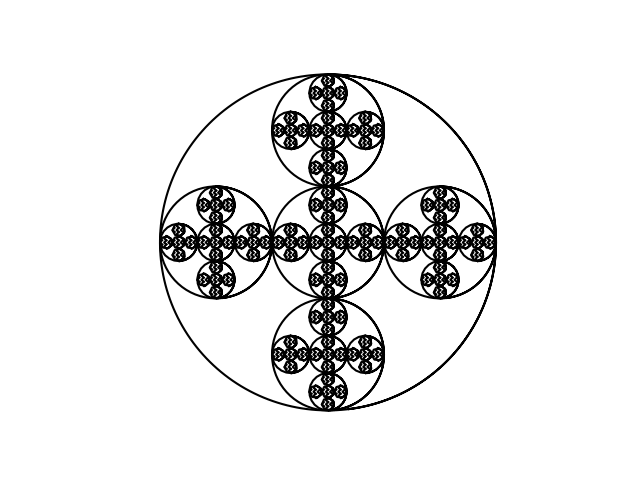
**T(n) = 5T(n – 1) + 1**

**Number of runtimes(n) = 3**



**T(n) = 5T(n – 1) + 1**

**Number of runtimes(n) = 4**



**T(n) = 5T(n – 1) + 1**

**Number of runtimes(n) = 5**

**CONCLUSION**

At higher value of n >1000, the recursive call failed to finish executing completely, with “RecursionError : maximum recursion depth exceeded in comparison.” Apparently, for the desired output to be achieved an appropriate base case has to be developed after which the recursive call is then made at a declining rate. It is necessary to find the accurate coordinates for each method call which then makes successive recursive calls to produce the correct shapes.

**Appendix**

Programmed by Olac Fuentes

import numpy as np

import matplotlib.pyplot as plt

def draw\_squares(ax,n,p,w):

if n>0:

i1 = [1,2,3,0,1]

q = p\*w + p[i1]\*(1-w)

ax.plot(p[:,0],p[:,1],color='k')

draw\_squares(ax,n-1,q,w)

plt.close("all")

orig\_size = 800

p = np.array([[0,0],[0,orig\_size],[orig\_size,orig\_size],[orig\_size,0],[0,0]])

fig, ax = plt.subplots()

draw\_squares(ax,15,p,.8)

ax.set\_aspect(1.0)

ax.axis('off')

plt.show()

fig.savefig('squares.png')

import matplotlib.pyplot as plt

import numpy as np

import math

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,w):

if n>0:

x,y = circle(center,radius)

ax.plot(x,y,color='k')

draw\_circles(ax,n-1,center,radius\*w,w)

plt.close("all")

fig, ax = plt.subplots()

draw\_circles(ax, 50, [100,0], 100,.9)

ax.set\_aspect(1.0)

ax.axis('off')

plt.show()

fig.savefig('circles.png')