**Introduction**

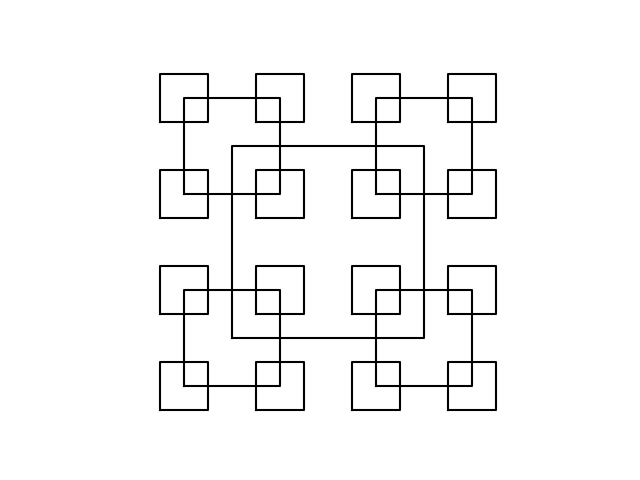
The problem is to draw figures that are shown in the lab assignment recursively. The first part of the figure is: each square on every corner of big square. The second part is: to draw circles inside of bigger circle but the left side of each circle will be nearly intersected. The third part of this section is: draw a binary tree. And the last part of is: to draw 5 circles on center, up, down, right, and left inside of a big circle. I will put more in detail about each section with explanation.

**Part 1**

The way I reach out first to this problem was to draw separate each edge of the big square to four different part. Before drawing the first edge, call a recursive method to draw a smaller square. However, after n > 2, it was difficult to find the next location of smaller square. Therefore, I changed to the idea of simply draw the big square and continue to the next four squares on the corner. Instead of thinking this problem as drawing a square, I did it as I was drawing a square. The parameter of the method, it contains (ax, c, r, n): c is the center of square, r is the radius, n is the number of recursive calls. As soon as the base case passes, array is created from using c and r. Each index is the location of corners as well as the center of the smaller squares. After drawing, the big square, recursive call will be made with new center location, halved radius, and (n-1). After some iterations are made, eventually, n reaches 0 and base case breaks.

n=2A close up of a logo

Description automatically generated

n=3

n=5 A close up of a piece of paper

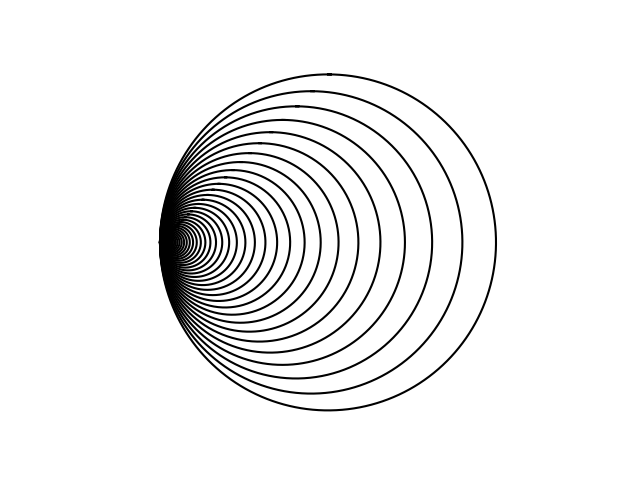
Description automatically generated

part 2

For this problem, I realized two things. First, circles are getting smaller as it moves to the left. Second, every y-coordinate of the center stays the same for all the circles. Since the circles are getting smaller, I knew the radius is changing at a certain rate depending on the figure. In addition, x coordinate moves to the left and all the circle intersects on the left-hand side of the circle. From this information, I knew that rad and x-coordinate are multiplied by unique number. Therefore, I have (ax, n, center, radius, w): n is the number of circles, center is the coordinates for the circle, radius, and w is the unique number which moves and make circles smaller.

W=0.6, n=10A close up of a logo

Description automatically generated

w=0.9, n=50

w=9.5, n=100A close up of a logo

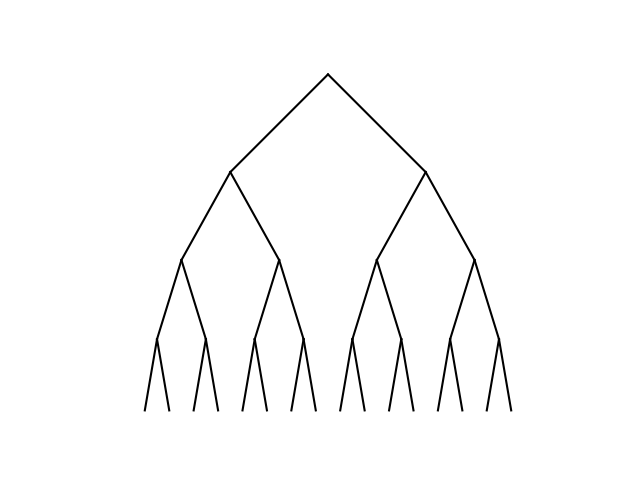
Description automatically generated

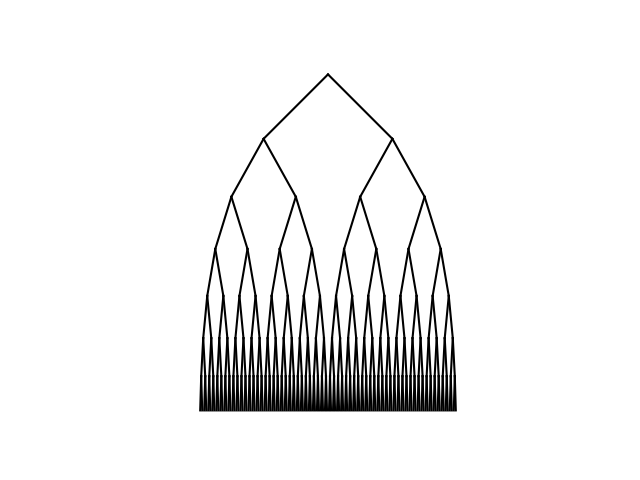
part 3

Since the goal figure is a binary tree, it requires to have two points to draw a straight line, and two recursive call from each root (and points). The root is set to [0,0] at first and next two points are one on the down left side and the other on down right. Therefore, both points’ y-coordinates are subtracted by some number (dy) and x is added or subtracted by some number (dx). In my case, dx and dy is set to 50, however, it will overlap the lines if dx is not changed. In order to avoid from happening this is to multiple dx by 0.5, and dy with 0.9 to make the tree look normal. Parameter for recursive method contains (ax, c, n, dx, dy): c as a root or the new principle point, n is number of heights of tree, dx and dy is numbers used to get the new c point or new root.

N=3A close up of a logo

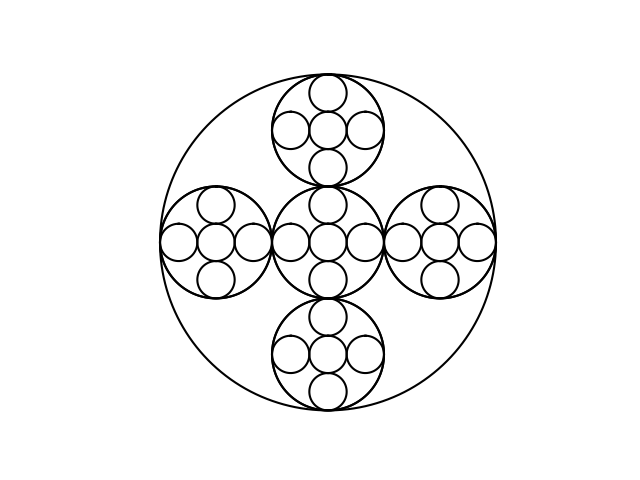
Description automatically generated

N=4

N=7

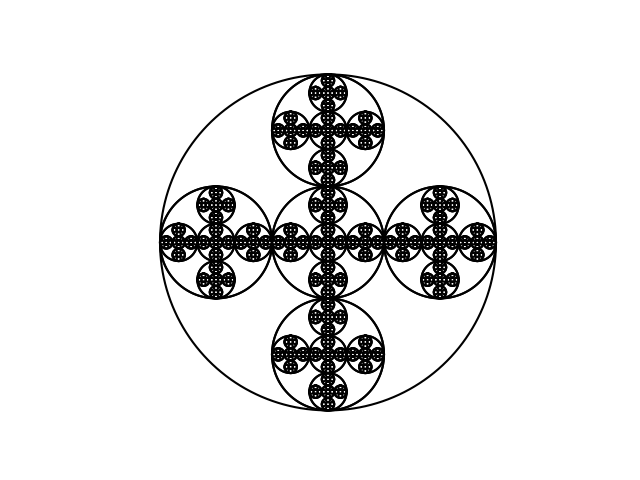
Part 4

The last part of the assignment is to draw 5 circles inside of a big circle. 5 circles must have the form of + which means one in the center, right, left, up, and down. From this information, it requires to have 5 recursive calls and draw 5 circles. The only thing that I had to be very careful is the way to get the center of the new circles and rad for the new circles. From the big circle, we know that diameter is 2r and 2r/6 gives us the new radius (newrad) for five smaller circles. After I determined the center, the only thing that is left to do is to find out five new centers. The difference between the big radius and new radius is the movement either to up-down or left-right. Adding difference to x-coordinate of the center gives the right circle and subtracting gives the left circle, adding to y-coordinate gives the up circle, subtracting gives the down circle. All the recursive calls send value with corresponded circle. Thus, in my parameter I have (ax, n, center, rad): n is the number of recursive calls needed, center of the circle, and radius of the circle.

N=2

N=3A close up of a logo

Description automatically generated

n=4

"""

Lab1 - All included.

2/9/2019 - Ken Amamori

CS2302 MW 10:30 - 11:50

"""

import matplotlib.pyplot as plt

import numpy as np

import math

def draw\_5circles(ax, n, center, rad):

if n>0:

newrad = rad/3 #new radius

circle(ax,[center[0]+rad-newrad,center[1]],newrad) #draw a circle: right

draw\_5circles(ax,n-1,[center[0]+rad-newrad,center[1]],newrad) #call a recursive method with new center and rad

circle(ax,[center[0]-rad+newrad,center[1]],newrad) #draw a circle: left

draw\_5circles(ax,n-1,[center[0]-rad+newrad,center[1]],newrad)#call a recursive method with new center and rad

circle(ax,center,newrad) #draw a circle: center

draw\_5circles(ax,n-1,center,newrad) #call a recursive method with same center and rad

circle(ax,[center[0],center[1]+rad-newrad],newrad) #draw a circle: up

draw\_5circles(ax,n-1,[center[0],center[1]+rad-newrad],newrad) #call a recursive method with new center and rad

circle(ax,[center[0],center[1]-rad+newrad],newrad) #draw a circle: down

draw\_5circles(ax,n-1,[center[0],center[1]-rad+newrad],newrad) #call a recursive method with new center and rad

circle(ax,center,rad) #big circle

#ax: , c: root, n: depths, dx,dy: difference from root to the child in x-, y-axis

def draw\_tree(ax, c, n, dx, dy):

if n>0:

ax.plot([c[0],c[0]-dx],[c[1],c[1]-dy], color='k') #draw a line from the root to the left child

ax.plot([c[0],c[0]+dx],[c[1],c[1]-dy], color='k') #draw a line from the root to the right child

draw\_tree(ax,[c[0]-dx,c[1]-dy],n-1,dx\*0.5,dy\*0.9) #call a recursive call with new root(left child) and change the deltaX & Y

draw\_tree(ax,[c[0]+dx,c[1]-dy],n-1,dx\*0.5,dy\*0.9) #call a recursive call with new root(right child) and change the deltaX & Y

#ax, center: center, rad: radius of a circle

def circle(ax,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)

ax.plot(x,y,color='k') #draw the circle

#n times, centre, radius, next circle line

def draw\_circles(ax,n,center,radius,w):

if n>0:

circle(ax,center,radius) #draw a circle with given center and radius

draw\_circles(ax,n-1,[center[0]\*w,center[1]],radius\*w,w) #call the method with new center and radius

#c: center of the square, r: radius, n: base case, n layers

def draw\_squares(ax,c,r,n):

if n>0: #base case: if n is greater than 0

p=np.array([[c[0]-r,c[1]-r], [c[0]+r,c[1]-r], [c[0]+r,c[1]+r], [c[0]-r,c[1]+r], [c[0]-r,c[1]-r]]) #get the four points of square based on 'Center' and 'r'

ax.plot(p[:,0],p[:,1],color='k') #draw a square with given array P

draw\_squares(ax, [c[0]-r, c[1]-r], r/2, n-1) #left bottom with halfed rad

draw\_squares(ax, [c[0]+r, c[1]-r], r/2, n-1) #right bottom with halfed rad

draw\_squares(ax, [c[0]+r, c[1]+r], r/2, n-1) #right above with halfed rad

draw\_squares(ax, [c[0]-r, c[1]+r], r/2, n-1) #left above with halfed rad

plt.close("all") #else a figure window

fig, ax = plt.subplots()

#for part 1

#calling a method "draw\_squares" with ax, center coordinate, radius, number of recursive methods

draw\_squares(ax,[500,500],500,2) #for 1a

#draw\_squares(ax,[500,500],500,3) #for 1b

#draw\_squares(ax,[500,500],500,4) #for 1c

#for part 2

#call a method called draw\_circles(ax, number of circles, centre, radius, interval)

#draw\_circles(ax, 10, [100,0], 100,.6) #for 2a

#draw\_circles(ax, 50, [100,0], 100,.9) #for 2b

#draw\_circles(ax, 100, [100,0], 100,.95) #for 2c

#part 3

c=np.array([0,0]) #starting root coordinate

#draw\_tree(ax, c, 3, 50, 50) #for 3a

#draw\_tree(ax, c, 4, 50, 50) #for 3b

#draw\_tree(ax, c, 7, 50, 50) #for 3c

#part 4

#a method named draw\_circles(ax, number of 5circles set, center, rad)

#draw\_5circles(ax, 2, [0,0], 100) #for 4a

#draw\_5circles(ax, 3, [0,0], 100) #for 4b

#draw\_5circles(ax, 4, [0,0], 100) #for 4c

ax.set\_aspect(1.0) #x:y=1:1

ax.axis('on')

plt.show() #dispaly figure

#fig.savefig('lab1\_2c.png')

**Conclusion**

I learned that by using recursion, a problem that seems to be hard or that require many lines of code, can be solved in a very efficient manner. Just by changing the number for n (number of recursive calls), it determines how many times will be repeated. In addition, I learned a lot about python since it has been one of the first time to use this language. I found very convenient and flexible how array works as well as the process of drawing a figure.

I certify that this project is entirely my own work. I wrote, debugged, and tested the code begin presented, performed the experiments, and wrote the report. I also certify that I did not share my code or report or provided inappropriate assistance to any student in the class.