Computer Science I
Circles

CSCI 141
Homework 3
08/29/20

1 Overview

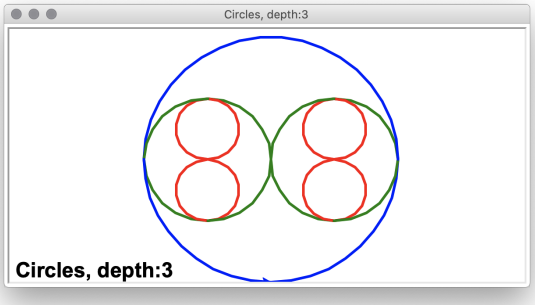
In this homework you will produce a non-recursive circle-drawing program that draws circles within circles within circles: *depth* 3. You will then write a general recursive circle-drawing program that can handle any depth.

2 Task 1: Non-Recursive Circles

You are provided with starter code in a file named `circles.py`. It is in the same directory in which you found this document. Download that file into a new Python project:

`https://www.cs.rit.edu/~csci141/Homeworks/03/circles.py`

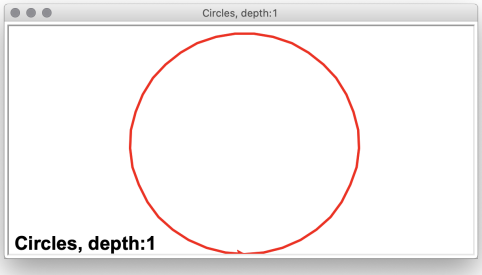
Fill in the functions that generate the circle pattern below. Details follow. Please note that while the images below show multiple colors and text labels, your program is not expected to do so.



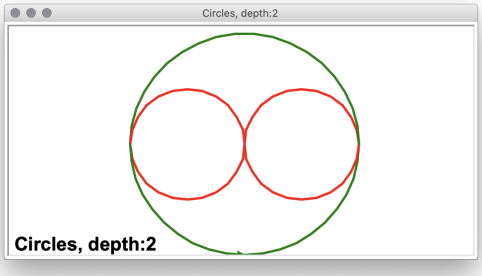
2.1 Program Operation and Requirements

You should write three separate, non-recursive functions.

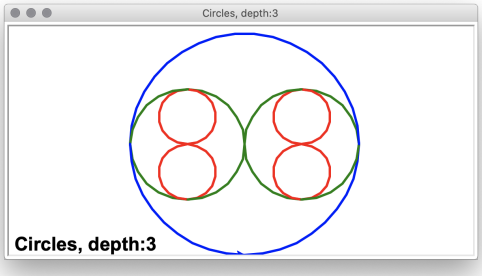
1. Write a function `draw_circles_1` that takes a `radius` as a parameter and draws the circle shown in red.



2. Write a function `draw_circles_2` that takes a `radius` as a parameter and draws the circle shown in green. **It must call `draw_circles_1` with half the original radius to draw the red circles.**



3. Write a function `draw_circles_3` that takes a `radius` as a parameter and draws the circle shown in blue. **It must call `draw_circles_2` with half the original radius to draw the green and red circles.**



2.1.1 Hints and Suggestions

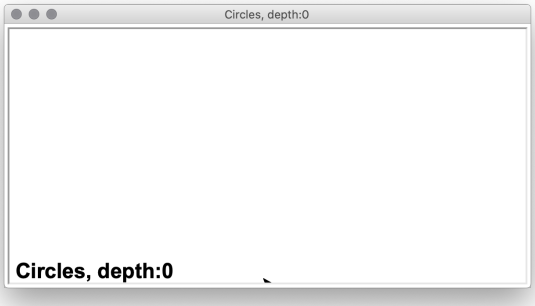
- Your program should be fully documented to receive full credit.
- There is an optional second argument to `turtle.circle()` that takes the *extent* of the circle to draw – the angle of an arc. By default this extent is `360` for specifying a full circle. A call of `turtle.circle(100, 90)` would draw the first quarter of a circle of radius `100`.
- Your program does not need to color the circles or place text alongside the drawings. But in case you are interested:
  - You can set the current drawing color using `turtle.color(`*`name`*`)`, where *`name`* is a string color name, like `'red'`, `'green'`, or `'blue'`.
  - Refer to the functions `title()` and `write()` in the turtle documentation: `http://docs.python.org/py3k/library/turtle.html`

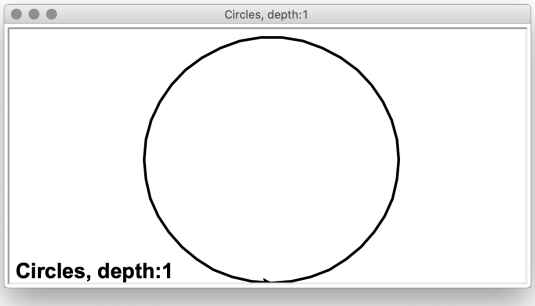
3 Task 2: Recursive Circles

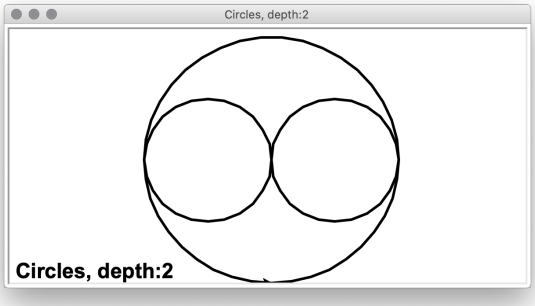
Fill in the function `draw_circles_rec(radius,depth)` in **circles.py** that generates the circle patterns shown below. Notice that the program prompts the user to enter the desired depth, a non-negative integer. This value controls the depth of recursion, i.e., how many different sizes of circles will be seen on the screen.

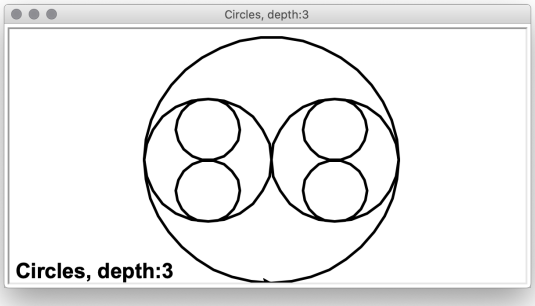
- If `depth` = 0, draw nothing.
- If `depth` = 1, draw a single full sized circle.
- If `depth` = 2, draw the same full-sized circle as for `depth` = 1. In addition draw two half-sized circles at the left- and right-most points of the full sized circle.
- If `depth` = 3, draw the same circles as for `depth` = 2. In addition draw four quarter-sized circles at the top and bottom-most points of the half-sized circles. [1]
- Higher values of `depth` will behave similarly.

You will not receive full credit if you do not use recursion to draw the pattern!









3.1 Program Operation and Requirements

Below is the way the console should look when the program is run. For this run, the user chose a depth of 2 for the recursive case. Note that the user has to hit the ENTER key a total of four times.

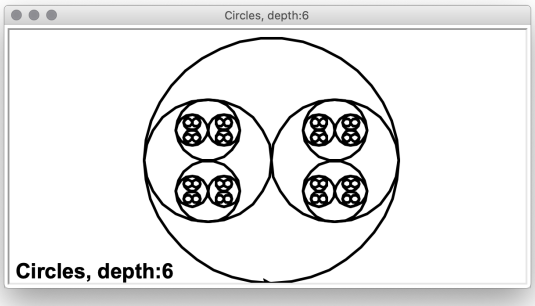
---

1. Because at each new depth the turtle is oriented at 90° from where it started the previous depth, the code will consistently draw the half circles at the same points.

```
Drawing a depth-1 circles drawing.
Hit ENTER to proceed to depth 2:
Hit ENTER to proceed to depth 3:
Hit ENTER to proceed to recursive code:
depth? 2
Close the window to end the program.
```

3.1.1 Hints and Suggestions

- Your recursive function `draw_circles_rec(radius,depth)` will need to take two arguments, the current radius and depth.
- Depth is determined by the user at run-time. As seen in the provided main function code, it must be an integer.
- The depth should be able to go beyond 3 in the same manner. Here is an example where $depth = 6$:



- Again, it is not required to title your window or draw text on the screen.
- Remember that your program should be fully documented to receive full credit.

4 Grading

- 30%: The non-recursive program functions correctly and contains the three separate drawing functions that call one another.
- 60%: The recursive function produces the expected picture for any reasonable depth value that does not cause the circles to be smaller than the pixel size on the display.
- -40%: A *loss* of points will occur if `draw_circles_rec(radius,depth)` is not recursive or uses more than one drawing function.
- 5%: Each function has a *docstring* containing its purpose, arguments (including type and description), and a list of pre and post conditions. Here is an example of such a docstring. Variations in format are allowed as long as the needed information present and formatting is reasonable.

```
def draw_tree (depth, length):
""" Recursively draw the tree shape.
    :param depth: (int) The current depth of recursion
    :param length: (int) The length of the current line segment/s
    Preconditions:
        depth >= 0, length > 0,
        turtle is facing north,
        turtle pen is up.
    Postconditions:
        Segment(s) of the tree were drawn for the current depth,
        turtle is facing north,
        turtle pen is up.
"""
```

- 5% The program is in a standard style, starting with a docstring for the whole file. This program file docstring must contain the assignment name, your name, and a purpose statement.

5 Submission

Zip your **circles.py** into a file called **hw03.zip** and submit that file to the mycourses drop box for this assignment.