Newton's Fractals

Function putValue(x, y, z0):

Input(s):

- x = the x position of the pixel in the image
- y = the y position of the pixel in the image
- z0 = the initial guess which is equivalent to the complex number of the tuple (x, y)

Output:

- does not return anything

Variable(s):

- z1 = the new estimation using newton's method
- TOL = tolerance between z0 and z1

For k an integer from 0 to n

Calculate z1 using newton's method

if (the absolute value of z1 minus z0 is less than TOL)

break from the loop

z0 is set to z1 to go through the process again to get closer to the root

puts the value of the 3-tuple (a color) in the tuple (x, y) in the image based on k in the loop

Main Code

Variable(s):

- Imgx = the number of pixels in the image horizontally
- Imgy = the number of pixels in the image vertically
- Image = the new image you are creating
- Top, bottom, left, and right = the length of the "axes" created in the image
- N = the max amount of iterations to find the root closest to the initial guess
- TOL = represents the tolerance at which we will stop the iterative process
- F = the original equation
- D = the derivative of the equation

Initialize imgx and imgy to 512

Initialize new image with the size of imgx by imgy

Initialize top, bottom, left, and right

Initialize n

Initialize TOL

Initialize f

Initialize d

for i an integer from 0 to 512

Variable: y = numerical value of a vertical position of the image

for j an integer from 0 to 512

Variable: x = numerical value of a horizontal position of the image

Variable: z0 = the complex number that refers to the position of the tuple (x, y)

Call putValue function

Image.save: Saves the image as a png under the name you wish to name the file.

I had a hard time with this program in the beginning. Graphing caused there to be no clear distinction between colors. I began to google and found the image module and found how to treat it as a file that can be generated from a program. I really enjoyed this program over all and have gained a great amount of python skills throughout this class. One of the challenges was determining how to convert a tuple into a number, but eventually found the complex function that allowed me to graph a corresponding color to each pixel in the image.