Assignment 2

Goals

The goal of this assignment is to work on control structures and functions in Python.

Instructions

You will be doing your work in a Jupyter notebook for this assignment. You may choose to work on this assignment on a hosted environment (e.g. [tiger](https://tiger.cs.niu.edu/jupyter/)) or on your own local installation of Jupyter and Python. You should use Python 3.8 or higher for your work. To use tiger, use the credentials you received. If you work remotely, make sure to download the .ipynb file to turn in. If you choose to work locally, [Anaconda](https://www.anaconda.com/download) is the easiest way to install and manage Python. If you work locally, you may launch Jupyter Lab either from the Navigator application or via the command-line as jupyter-lab.

In this assignment, we will be working on programs related to John Conway’s [FRACTRAN](https://en.wikipedia.org/wiki/FRACTRAN) language. The language has rules based on a list of fractions where steps are based on whether the the multiplication of an integer by the fraction produces an integer. The programs created by this language leverage prime factorizations of numbers. In addition, FRACTRAN can be used to generate prime numbers and the digits of pi. In fact, it is Turing-complete.

Due Date

The assignment is due at 11:59pm on **Monday, September 13**.

Submission

You should submit the completed notebook file required for this assignment on [Blackboard](https://webcourses.niu.edu/). The filename of the notebook should be a2.ipynb.

Details

Please make sure to follow instructions to receive full credit. Use a markdown cell to **Label** each part of the assignment with the number of the section you are completing. You may put the code for each part into one or more cells.

0. Name & Z-ID (5 pts)

The first cell of your notebook should be a markdown cell with a line for your name and a line for your Z-ID. If you wish to add other information (the assignment name, a description of the assignment), you may do so after these two lines. Because we are concentrating on control flow and functions, do not use lists, other collections, or comprehensions for this assignment. Lists are allowed for the extra credit.

1. Logarithm Base 3 (15 pts)

We have seen that Python has an exponentiation operator (\*\*), and the inverse of exponentiation is the logarithm. Specifically, if c = a \*\* b, the logarithm of c, base a is b:b=loga⁡cWhile Python has a logarithm method in the math library, we will be building our own for thee logarithm base 3. To compute this, count the number of times that you can repeatedly divide a number by 3 until you reach 1. For example, 27 / 3 = 9, 9 / 3 = 3, and 3 / 3 = 1. Thus, log3⁡27=3. For now, we will be happy with an **integral** result so we will count the number of times until we are below 1, and then subtract 1 from the final result. So 10 / 3 = 3.333, 3.333 / 3 = 1.111, and 1.111/3 = 0.370 < 1 so our answer is 3 - 1 = 2. Use a while loop to perform the division. Here is ok to use a variable to count the number of divisions.

Use your log function to compute the results for 27, 81, and 2187. The results should be 3,4, and 7.

2. Addition in FRACTRAN (25 pts)

Now, we will use our new logarithm function to create a FRACTRAN addition function named fractran\_add. The FRACTRAN program for addition is the single fraction (3/2). It takes an input 2a3b and produces an output 3^{a+b}. Write a function that does FRACTRAN addition. It should take a and b as input and then

1. Calculate n=2a3b
2. Loop as long as n is even
   1. Multiply n by 3 and divide by 2 and store the result in n.
3. Compute the logarithm (base 3) and return the result.

As an example, assume a=4, b=3. We compute n=2433=432. It is even, so we have 432 \* 3 / 2 = 648. We continue and we get 972, 1458, and then 2187. 2187 is not even and thus cannot be divided by 2. So we apply our log function and get log3⁡2187=7, which is 4 + 3! Seems like a long way to create an addition function, but note that we don’t use the addition operator except to count in the logarithm function.

Write code to run your function for all combinations of values of a from 1 to 5 and values of b from 3 to 7. Print the results for each combination.

Hints

* If you have not completed Part 1, you can use import math and math.log(n, 3) to compute the log base 3 of n
* Checking whether a number is even is the same as checking whether its remainder when divided by two is zero
* Use a nested for loop to print all the results.
* Remember to use range to generate sequences of numbers.

3. Subtraction in FRACTRAN (25 pts)

Subtraction in FRACTRAN is a little more complicated. This time we will use the FRACTRAN program 2/15, 1/2 (two steps) and start with numbers of the form 3a5b and compute 3a−b. Here, we have the steps:

1. Calculate n=3a5b
2. Loop as long as n is divisible by 15
   1. Multiply n by 2 and divide by 15 and store the result back in n.
3. Loop as long as n is divisible by 2
   1. Divide n by 2 and store the result back in n
4. Compute the logarithm (base 3) and return the result.

Write the fractran\_sub function to compute subtraction according to the FRACTRAN algorithm above. Use the function from Part 1 to complete step 4. Print the results for all combinations of values of a from 5 to 10 and values of b from 1 to 5.

Extra Credit

Write a general FRACTRAN interpreter that takes a list of fractions and computes the result. See the [definition](https://en.wikipedia.org/wiki/FRACTRAN). Use this interpreter to test the addition and subtraction programs above (3/2), (2/15, 1/2), respectively.