Programming Problems

For each of these problems (unless otherwise specified), write the needed code directly in the Python file in the corresponding function definition.

All programming problems may also be checked by running 'Run current script' on the starter file, which calls the function `testAll()` to run test cases on all programs. Before submitting, make sure your code runs without raising a syntax error; any syntax errors remaining in the code will lead to a deduction in your overall grade. Runtime and assertion errors will also affect the grades of individual problems. You should check your code this way for all future programming assignments as well.

#1 - `numSign(x)` - 10pts

Can attempt after Booleans, Conditionals, and Errors lecture

Write a function **numSign(x)** that takes a number as a parameter and **returns** a string representing its sign. More specifically, the function should return `"positive"` if the number is positive, `"negative"` if it is negative, and `"zero"` otherwise.

For example, `numSign(12)` should return `"positive"`, `numSign(-0.5)` should return `"negative"`, and `numSign(0)` should return `"zero"`.
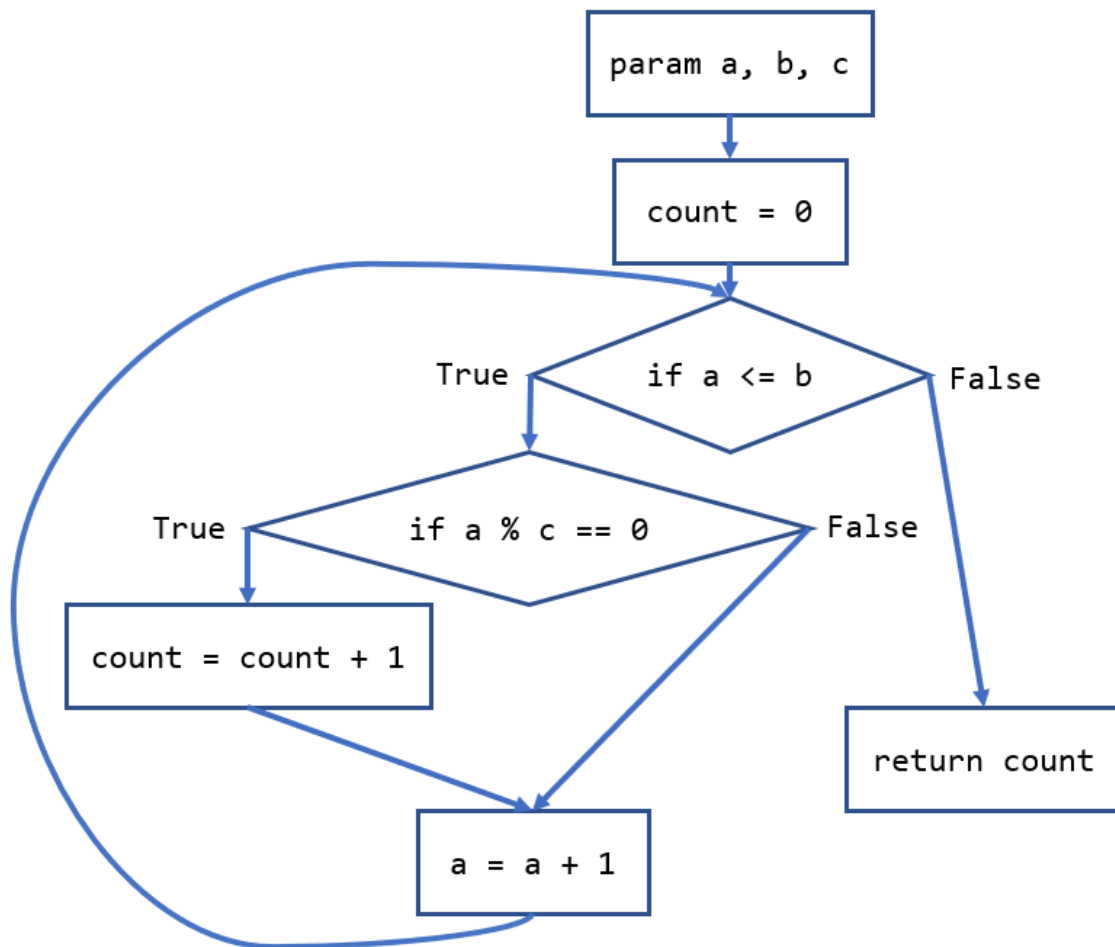
You are guaranteed that the function will only be called on ints and floats.

#2 - Flow Chart to Program - 15pts

Can attempt after While Loops lecture

Given the control flow chart below, write a function `mysteryFunction(a, b, c)` that implements the control flow chart correctly.

Note: make sure to run the test cases to see if you've implemented the code correctly! If the test case fails or your code loops infinitely, check whether your indentation matches what is expected here.

```
                              param a, b, c

                                count = 0

            True          if a <= b          False

     True         if a % c == 0       False

count = count + 1

                                                  return count

                    a = a + 1
```

#3 - Interactive Program - 15pts

Can attempt after Booleans, Conditionals, and Errors lecture

In the function **interactiveProgram,** use the `input` function and conditionals to set up a short interactive program of your own design. This could be a very short choose-your-own-adventure story, or a Buzzfeed-style quiz, or whatever else you'd like! The only requirements are:

1. You must use the `input` function to collect information from the user at least three times.
2. The `interactiveProgram` function should take no parameters.
3. You must use **conditionals** somewhere in your code. There should be at least three `if`, `elif,` or `else` statements.
4. All the code for your interactive program must be in the `interactiveProgram` function (or functions that `interactiveProgram` calls). Do not put calls to `input` at the top level - this will break the autograder.