15-110 Check2 - Written Portion

Name:

AndrewID:

Complete the following problems in the fillable PDF, or print out the PDF, write your answers by hand, and scan the results.

When you are finished, upload your check2.pdf to **Check2 - Written** on Gradescope, and upload your check2.py file to **Check2 - Programming** on Gradescope. Make sure to check the autograder feedback after you submit!

Written Problems

#1 - Evaluating Boolean Expressions - 5pts

Can attempt after Booleans, Conditionals, and Errors lecture

For each of the following Boolean expressions, determine whether it evaluates to `True`, `False`, or an error.

```
(4 > 5) and ("foo" == "foo")
```
☐ True
☐ False
☐ Error

```
(10 > 0) or (0 == 1/0)
```
☐ True
☐ False
☐ Error

```
not (True and False)
```
☐ True
☐ False
☐ Error

```
(2 <= 5) and (4 + "a" == "4a")
```
☐ True
☐ False
☐ Error

```
("a" == "A") or (0 > 1)
```
☐ True
☐ False
☐ Error

#2 - Code Tracing Conditionals - 15pts

Can attempt after Booleans, Conditionals, and Errors lecture

Given the following block of code, choose specific values for x, y, and z that would lead to the code printing A, B, C, D, or E. If one of the variables could be assigned to any value to achieve the result, write the word **anything** instead of a value. Fill out your answers in the table below.

```python
if x < 10:
    if y > 20:
        if z == "foo":
            print("A")
    else:
        if y % 2 == 0:
            print("B")
        else:
            print("C")
elif x < 100:
    if y < 0 and z == "bar":
        print("D")
    elif y < 0:
        print("E")
```

Printed Result	x value	y value	z value
A			
B			
C			
D			
E			

#3 - Circuit to Truth Table and Expression - 22pts

Can attempt after Circuits and Gates lecture

Fill out the truth table below to show all possible input combinations and all resulting outputs for the circuit shown here. You may not need to use all the rows.

x value	y value	z value	output value

Write a Boolean expression which is equivalent to the circuit above in the box below.
Note: it's fine to use normal Boolean terms (and/or/not/xor) instead of circuit operators.

#4 - Code Tracing While Loops - 18pts

Can attempt after While Loops lecture

Given the following block of code, fill out a variable table that shows the values of the variables at the **end** of each iteration of the loop. You may not need to fill out values for every listed iteration.

```
x = 0
y = 10
z = 0
while x <= y:
    x = x + 3
    y = y + 1
    z = (x + y) - z
    print(x, y, z)
```

	x value	y value	z value
Pre-loop	0	10	0
Iter 1	3	11	14
Iter 2	6	12	4
Iter 3	9	13	18
Iter 4	12	14	8
Iter 5	15	15	22
Iter 6	18	16	12
Iter 7			
Iter 8			

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 File As 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 an error message. Any syntax or runtime errors left in the code will result in a deduction on the assignment grade. You should do this 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.
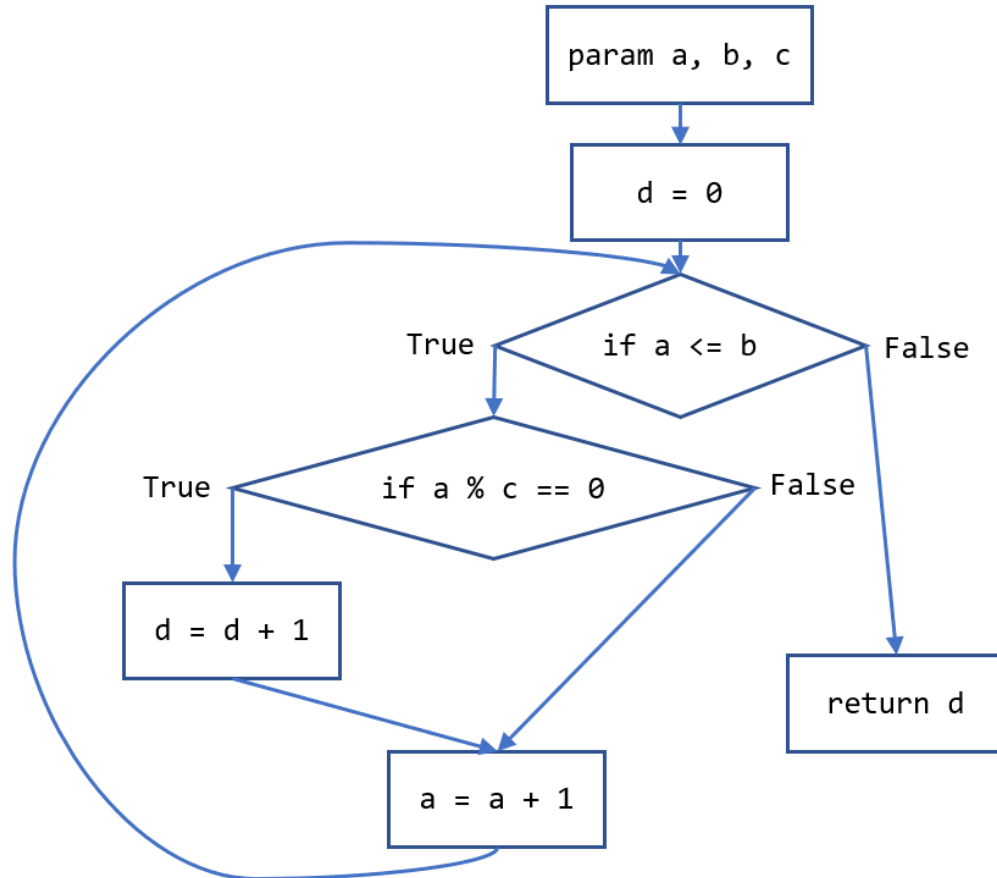
#3 - Interactive Program - 15pts

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 two `if` statements and at least one `elif` or `else` statement.
4. All the code for your interactive program must be in the `interactiveProgram` function (or helper functions that `interactiveProgram` calls).

Note: we won't officially cover `input` until the While Loops lecture