CSC 110
CSC110 Intro to CS through programming
Homework 01

Spring 2025

Variables and Operations

`Due: By next Friday, Midnight`

Note: Homework Assignment 1 should be completed individually.

Instructions

Follow the instructions for parts 1 through 4. Note that at the end of the instructions there is a section reminding you where stuff is and how to run the tests.

READ all instructions before starting! Notes :

- All your editing will happen in the file called hw01.py.
- As you advance, make small edits and then run the program, in a Bash shell, with instruction:

```
python3 hw01.py
```

- Deal with any issues (syntax errors, runtime errors, or logical errors) and then continue.

Part 1

Add code to the part 1 section after the line:

```
# Your code for part 1 under this line and before the print statements
```

and before the comment:

```
# End of Part 1 ----------------------
```

Overwrite the required variables so that you complete the following steps (in python):

- assign 27 to a variable named x
- assign 1 to a variable named y
- assign 1.5 to a variable named a
- assign 7 to a variable named b
- assign -1 to a variable named c

Note that you do not modify the initial assignments at the top, where all variables are initialized with zeros. Instead you need to overwrite their values with the ones indicated above.

Then, you need to write an arithmetic expression, in python so you obtain the following:

$result1 = \frac{3x-9y}{2a(b-c)}$

At the end, the print statement should print:

```
Part 1: result = 3.0
```

Do not just assign the value of 3.0 directly to result1!

There are tests in the test section to check if you did this correctly (more on tests at the end of these instructions)

Part 2

Add code to the part 2 section after the line:

```
# Your code for part 2 under this line and before the print statements
```

and before the comment:

```
# End of Part 2 ----------------------
```

Overwrite the required variables so that you complete the following steps (in python):

- assign 5 to a variable named x
- assign -3 to a variable named y

Note that you do not modify the initial assignments at the top, where all variables are initialized with zeros. Instead you need to overwrite their values with the ones indicated above.

Then, you need to write an arithmetic expression, in python so you obtain the following:

$result2 = x^2y^4$

At the end, the print statement should print:

```
Part 2: result = 2025
```

Do not just assign the value of 2025 directly to result2!

There are tests in the test section to check if you did this correctly (more on tests at the end of these instructions)

Part 3

We need to solve a problem of "fair distribution". Let's say you have 100 treats and 13 dogs. Now, we want to give an equal number of COMPLETE treats to each dog. Here are the steps to perform the calculations.

Add code to the part 3 section after the line:

```
# Your code for part 3 under this line and before the print statements
```

and before the comment:

```
# End of Part 3 ----------------------
```

Overwrite the required variables so that there is one line of code for each of the following steps:

- assign 100 to a variable named a (representing the total number of treats)
- assign 13 to a variable named b (the total number of dogs)

Then, you need to write an arithmetic expression, in python so you obtain the following:

$result3$ = the number of whole (integer) treats that each dog gets

At the end, the print statement should print:

```
Part 3: result = 7
```

Do not just assign the value of 7 directly to result3!

There are tests in the test section to check if you did this correctly (more on tests at the end of these instructions)

Part 4

We need to solve a problem of calculating leftovers. For the problem solved above (100 treats split evenly into 13 dogs, without breaking up treats), there will be a leftover. We need to calculate this amount using Python.

Add code to the part 4 section after the line:

```
# Your code for part 4 under this line and before the print statements
```

and before the comment:

```
# End of Part 4 ----------------------
```

Overwrite the required variables so that you complete the following steps (in python):

$result4$ = the number of whole (integer) leftover treats that we have after giving each dog an equal number of treats.

Note : There is no need to reassign a and b, we'll use the previous values (from part 3) Hint : Use Modulo!

At the end, the print statement should print:

```
Part 4: result = 9
```

Do not just assign the value of 9 directly to result4!

There are tests in the test section to check if you did this correctly (more on tests at the end of these instructions)

Adding one instruction to the program

The Python program has a a header section you should fill in (replacing the placeholder info). The code has comments indicating where to add your code. Remember, "comments"... are text that do not get executed.

Expected Output

This is what the output should look like:

```
Part 1: x = 27
Part 1: y = 1
Part 1: a = 1.5
Part 1: b = 7
Part 1: c = -1
Part 1: result = 3.0
Part 2: x = 5
Part 2: y = -3
Part 2: result = 2025
Part 3: a = 100
Part 3: b = 13
Part 3: result = 7
Part 4: result = 9
```

Tests and checking your work

In addition to the hw01.py file, where you will make edits to complete the homework, you might see additional python files called test_<something>.py. You are not to modify these files (you don't even need to look at it). These files are provided so you can see how many tests you have passed so far. If you pass all tests, then your code is complete.

To run the tests, there are two things you can do:

- Go to the Tests icon on the left menu (looks like a triangular flask), and look for the test you want to run on the set of tests (the top level one runs them all). If you hover your mouse next to the test name, three play buttons will be displayed, the leftmost runs the test. Whe you do so, the output is shown on the lower-right window under the "TEST-RESULTS" tab.
- Alternatively, go to the terminal end run the following command (which will display all of the test output to the TERMINAL tab itself:

```
pytest -s -v
```

Grading criteria:

`General`

The submission:

- runs without syntax errors (or -50%)
- adds a few small but informative comments (or -10%)

`Operations`

The program:

- Passes all 17 tests (or lose 5% per missed test).

Submitting

I will collect all of your hw01 directory files at the due date/time so make sure it is complete and running by then.

Interpreting Test output

Check the slides in Lecture05. It has a visual explanation. In the TEST RESULTS window, under a long block if information that starts with `Running pytest with args:` You might see something like this (I added errors on purpose for educational purposes)

```
collected 17 items

test_hw01.py .........FF......                                           [100%]

================================== FAILURES ===================================
____________________________ test2_result2_printout ____________________________

capsys = <_pytest.capture.CaptureFixture object at 0x105af2450>
monkeypatch = <_pytest.monkeypatch.MonkeyPatch object at 0x105b64250>

    def test2_result2_printout(capsys, monkeypatch):
        global captured
        import hw01  # Import the module here
        # captured = capsys.readouterr()
>       assert "Part 2: result = 2025" in captured.out, "Tip: check the printout for part2's result is EXA
E       AssertionError: Tip: check the printout for part2's result is EXACT
E       assert 'Part 2: result = 2025' in 'Part 1: x = 27\nPart 1: y = 1\nPart 1: a = 1.5\nPart 1: b = 7\nl
E        +  where 'Part 1: x = 27\nPart 1: y = 1\nPart 1: a = 1.5\nPart 1: b = 7\nPart 1: c = -1\nPart 1: 

test_hw01.py:60: AssertionError
_____________________________ test2_result2 ______________________________

    def test2_result2():
        global captured
        import hw01  # Import the module here
>       assert hw01.result2 == 2025, "Tip: Did you assign the value of result2 correctly?"
E       AssertionError: Tip: Did you assign the value of result2 correctly?
E       assert 2026 == 2025
E        +  where 2026 = <module 'hw01' from '/Users/pfrank/Library/CloudStorage/Dropbox/Mac/Documents/csc

test_hw01.py:66: AssertionError
========================== short test summary info ===========================
FAILED test_hw01.py::test2_result2_printout - AssertionError: Tip: check the ...
```

```
FAILED test_hw01.py::test2_result2 - AssertionError: Tip: Did you assign the ...
========================= 2 failed, 15 passed in 0.05s =========================
Finished running tests!
```

This is the summary:

```
collected 17 items

test_hw01.py .........FF......                                           [100%]
```

It indicates there were 17 tests, of which 15 passed (".") and two failed ("F").

Below that, there are blocks of output for every test that failed.

Failed test 1

A test called "test2_result2" failed. Look for this line:

```
E       AssertionError: Tip: check the printout for part2's result is EXACT
```

It's hard to see but, the following was expected: "Part 2: result = 2025" but in the printout no such line is found. There is one that is close but incorrect: "Part 2: result = 2026"

Failed test 2

For the second test, the information states the following:

```
E       AssertionError: Tip: Did you assign the value of result2 correctly?
E       assert 2026 == 2025
```

As you can see, these two are related... your calculation for result2 is off. Once you fix that, you should pass all tests.