**IT FDN 100A: Foundations of Programming: Python**

**Instructor: Randal Root**

**Student: Morgan Lang**

**June 28, 2016**

# Assignment 2

## Assignment objectives:

1. Write a simple interactive Python program that transforms user input strings into number values, then performs simple arithmetic on those values.
2. Test the program.
3. Paste the resultant code into a Word document in which you describe your learning experience.
4. Submit the Word document via Canvas.

## Overview

This was a fun assignment, but I’m looking forward to upcoming modules dealing with if/then/else/elif logic. For example, for this assignment, it would have been useful to allow the user to select which arithmetic operations to perform. Also, it would be helpful to be able to restart the program in addition to quitting it. I know we’ll be covering those topics soon.

I have commented the code to explain individual steps.

## The program

I have tried to incorporate as many of the techniques discussed in the readings as seemed reasonable given the humble goals of this script. As in the previous assignment, I do strongly prefer to use the **time** module to delay certain parts of the script for a better interactive experience.

My method is pretty straightforward: I incorporate things that seem fun as I read about them, then refine them later. Sometimes, this method can lead me down a dead end. For this program, I got ahead of myself and initially made a classic mistake by trying to perform calculations on strings before transforming them into numbers. Once I learned about **int**, I then used it redefine user input strings as integers, but then realized that **float** would be better.

Instead of transforming user input (**askNumber1** and **askNumber2**) into two additional variables (**x** and **y**) and then defining those new variables as floats using this method:

**x = float(askNumber1)**

**y = float(askNumber2)**

I probably should have simply defined the user input as a float when prompting for it, like this:

**askNumber1 = float(input("Enter a number: "))**

But I decided to leave that unnecessary step in there as a reminder to myself that it is possible to define variables using other variables.

**Number-crunch-o-matic.py:**

**#Program name: number-crunch-o-matic.py**

**#Program function: to perform simple arithmetic on user-entered data**

**#Author: Morgan Lang**

**#Date modified: 06/28/2016**

**#################################################################**

**#First, we'll import the time module so we can control the timing of steps**

**import time**

**#Program introduction & credits using backslashes to demonstrate escape characters.**

**print ("\t\t\t \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\")**

**print ("\t\t\tSuper Amazing Number Crunch-O-Matic")**

**print ("\t\t\t \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\")**

**print ("\t\t\t\t\tby")**

**print ("\t\t\t\tMorgan Lang")**

**print ("\t\t\t \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\")**

**time.sleep(1)**

**#Although it's totally unnecessary for the purposes of this class, I believe that pacing is an important part of interactivity. This section creates the illusion of beginning the "real" part of the program and helps provide structure to the user.**

**print ("\n\n")**

**print ("\n\nHang on a sec.")**

**time.sleep(1)**

**print ("\nOK. Beginning program...")**

**#Assigning the return value of the input function to a variable named askNumber1.**

**askNumber1 = input("\nAll righty, then. Let's perform some simple arithmetic on two numbers. \nTo start, give me a number. You can use decimals and negative numbers. ")**

**time.sleep(1)**

**print ("\nOK. You entered", askNumber1, ". That's great. Now I'd like you to enter another number. ")**

**#Assigning the return value of another input function to a variable named askNumber2**

**askNumber2 = input("\nEnter a second number, please. ")**

**time.sleep(1)**

**print ("\nGreat. You entered", askNumber2, " . Now I'm going to add these two numbers. Hang on while I think for a sec.")**

**#Pausing for dramatic effect**

**time.sleep(1)**

**#This next step redefines the return values of askNumber1 and askNumber2 as floating point numbers. This step is necessary in order to perform mathematical operations, because values obtained using input() are, by default, strings. Attempting to perform addition on strings will concatenate them rather than add them.**

**#Note that this step wouldn't be necessary if I had defined the return values for askNumber1 and askNumber2 as floats when I first assigned values. For example:**

**#askNumber1=float(input("\nAll righty, then. Let's add two numbers. To start, give me a number. "))**

**#Redefining the return values of askNumber1 and askNumber2 as floats:**

**x = float(askNumber1)**

**y = float(askNumber2)**

**#Addition**

**print ("\nOK, got it. The sum of ", x, "and ", y, "is ", x+y,".")**

**time.sleep(1)**

**#Subtraction**

**print ("\nNow I'm going to subtract", y, "from", x, ". ", "Hang on while I think for a sec.")**

**time.sleep(2)**

**print ("\nOK, got it.", x, " minus", y, "is ", x-y,".")**

**time.sleep(1)**

**#Multiplication**

**print ("\nNow I'm going to multiply", y, "and", x, ". ", "Hang on while I think for a sec.")**

**time.sleep(2)**

**print ("\nOK, got it.", x, " multiplied by", y, "is ", x\*y, ".")**

**time.sleep(1)**

**#Division**

**print ("\nNow I'm going to divide", y, "by", x, ". ", "Hang on while I think for a sec.")**

**time.sleep(2)**

**print ("\nOK, got it.", y, " divided by", x, "is ", y/x,".")**

**time.sleep(1)**

**#Division**

**print ("\nFinally, let's find the average of", x, "and", y, ". ", "Hang on while I think for a sec.")**

**time.sleep(2)**

**print ("\nOK, got it. The average of", x, " and", y, "is ", (x+y)/2,".")**

**time.sleep(1)**

**#Exit prompt**

**input ("\nWe should totally do this again sometime. \nFor now, press the Enter key to quit this program.")**

**#Ring system bell when user responds**

**print("\a")**

**#Show some snazzy ASCII art for 2 seconds using blockquotes**

**print (**

**"""**

**\_\_\_\_ \_**

**| \_ \ | |**

**| |\_) |\_ \_ \_\_\_| |**

**| \_ <| | | |/ \_ \ |**

**| |\_) | |\_| | \_\_/\_|**

**|\_\_\_\_/ \\_\_, |\\_\_\_(\_)**

**\_\_/ |**

**|\_\_\_/**

**"""**

**)**

**time.sleep(2)**

I used this ASCII art generator to create the graphic farewell:

<http://patorjk.com/software/taag/#p=display&f=Big&t=Hello%20world>

## Miscellany:

I started out using Brackets for this assignment, then switched to PyCharm once it started crashing so I could debug it. PyCharm immediately determined that I was missing a single comma in one of my user prompts. It obviously is better to use a Python-specific IDE.