CSCA20H Lab 1 (Week 2)

Congratulations on getting started with A20 labs. Now it's time for you to do some programming. To earn your lab marks, you must submit your work by Saturday at 11.59pm.

Go to your lab or office hours if you're feeling lost!

Objectives

1. Practise working with expressions in the shell
2. Practise submitting a lab/exercise/assignment
3. Get Wing and Python working on your laptop (optional)

Numerical expressions in the shell

Now that the administrative details are out of the way, we can get started with the fun part of the lab! Many of you will be programming for the first time; that's exciting, but may also be a bit overwhelming. Don't hesitate to ask your TA or your labmates for help. You may also refer to your notes.

To begin, click the WING IDE icon in the “Start” menu. In this lab, we'll start by working in the Python shell. Your TA can demonstrate using the Python shell in the Wing IDE if you need it.

Python has many calculator-like features. The following is a list of expressions:

- 10
- 34 + 8
- 29 / 3
- 29 // 3
- 18 - 2.9
- 24 % 5
- 5.5 * 4.0
- 28 / 3.0
- 28 // 3.0
- 14 * 3 + 2
- 14 * (3 + 2)
- 1 / 0
- 1 // 0
- 1 % 0

For each expression, think about what you expect to see when you evaluate it. Then, evaluate the expression in the shell (or use the print() function within a .py file to print it) and see if Python agrees with you. Ask your TA if you need help or you don't understand something.

Types in the shell

Let's look at determining the types of some values and expressions. In Python, one determines the type of something by calling the type() function on it.

Try each of the following evaluations in the shell:

- type(1)
- type(3.14)
- type(1.0)
- type(-2)
- type(-2.0)
- type('hello world')

You may or may not have seen the 'float' type before. In Python, a float is a type of number which may or may not have an integer value. For example, one would use a float to represent the number 3.14. Note that one can also represent integers as floats eg. the number 1.0 is a float.

Built-in functions in the shell
Just as in math, Python functions take in a value and do something (hopefully useful) with it. You've already seen and played around with the print() and type() functions. These functions are provided by Python; they are called built-in functions. Let's play around with some more built-in functions!

The int() function tries to convert whatever you give it to an int (the type used to store integer numbers) if possible. If it needs to round a float, it rounds *down.*

Here are some expressions:
- int(3)
- int(3.14)
- int(2.8)
- int('4')
- int('hello world')

For each expression, think about what type of value we're passing int(). Then, think about what the most sensible outcome of the conversion would be. Then run the expression in the shell and see whether Python agrees with you! Then, try passing the resulting value into the type() function and check whether the value was indeed converted into an int.

The float() function is like the int() function, except it converts to - you guessed it - floats. Do the same as what you did with int() for the following expressions:
- float(3)
- float(3.14)
- float('4')
- float('3.14')
- float('hello world')

The max() and min() functions give you the maximum and minimum (respectively) of all the values that you give them. These will probably be the first functions you've seen which take multiple values instead of just one!

Try out these expressions:
- max(3.0, 1)

- max(-1, -2, -3)
- max(0, 'this is a string')

Then, for each expression, use min() instead of max().

The round() function rounds to the nearest integer. Try out the following, and also examine their types:

- round(3.14)
- round(2.8)
- round(0)
- round('supercalifragilisticexpialidocious')

If you'd like to learn more about these functions, you can look them up in the official Python 3 documentation https://docs.python.org/3/library/functions.html

Practise submitting an assignment

This section is very important!! If you leave figuring out how to submit your work until you really need to do it, you won't be able to, because that's how the world works.

The assignments in this course will be submitted electronically. You will need to go to Markus by visiting this website: https://markus.utsc.utoronto.ca/csca20f18 to submit your labs, exercises and assignments.

First, create a program that prints the string 'Hello world!'. Save this program as hello_world.py. It is important that you use the *exact* filename; make sure you use the correct case!

Then, log into Markus at the website above. Select the assignment "lab1: lab1" and submit your file hello_world.py.

In order to earn credit for this week's lab, you must complete this submission.

Installing Wing and Python on your own laptop (optional)

If you brought your own laptop to the lab and you haven't gotten Wing and Python installed and running on it yet, now's a good time to do so.

Here is the process you should follow to install Wing and Python:

1. Download the Python 3.7 installer from here: https://www.python.org/
2. Run the installer. Make sure to take note of where you installed Python because you might need it at a later step.
3. Download the Wing IDE 101 installer here: https://wingware.com/downloads/wingide-101 and run the installer
4. Run Wing IDE 101. Look at the contents of the shell (the sub-window at the bottom right). If the first line starts with "3.7.0", you're in luck! Wing IDE 101 has automatically found the correct version of Python for you and you don't need to do anything else.

5. If you see a different version, you need to point Wing to the correct Python installation. To do so, go to Edit->Configure Python->Python Executable. Choose the Custom option and then browse to the folder in which you installed Python 3.7 at step 2. Within that folder, select "python.exe".
6. Press the OK button at the bottom. You may be prompted to restart the Python shell; click yes.
7. The first line of the text in the Python shell should now start with "3.7.0"
8. If anything goes wrong or you get lost, feel free to ask your TA to help.