PIC 16, Winter 2018 – Preparation 1W

Assigned 1/8/2018. To be completed before lecture 1/10/2018.

Intended Learning Outcomes

By the end of this preparatory assignment, students should be able to:

- install the Anaconda distribution of Python on their personal computers, if desired;
- interact with the Python interpreter via command line, Jupyter Notebook, and Spyder IDE; and
- write Python programs consisting of numerics, strings, lists, and operators.

Tasks

- ☐ Carefully read the syllabus, “Regarding Tracks…”, and “Regarding Quizzes…”, and Lecture 1M slides if you haven’t already.
- ☐ (Optional) Install Python (Anaconda Distribution)
    - Visit https://www.continuum.io/downloads
    - Download Python 2.7 version (Not Python 3.6)
    - Run file and follow prompts. Allow Python folder to be added to the system path.
    - Note that three important programs have been installed: Python itself, Jupyter, and Spyder.
- ☐ Getting started with Python
    - (Optional) See video playlist “Getting Started with Python” in Week 1 of CCLE
    - Using the command line.
        - On Windows, select the start menu and search “cmd”. This should open a command prompt in a console window.
        - Type `python` and press enter. You should get a welcome message and your prompt should change to `>>>`. If you get an error, the Python folder was not added to the system path. Search Google for “add python to system path Windows 10” or similar for solutions.
        - Type `print “Hello World!”` and press enter. The string should be echoed below your command. Try some arithmetic and see what happens.
    - Using a Jupyter Notebook
        - Start the program “Jupyter Notebook”. A console window will open and so will a tab in your default browser. *Do not close the console window.*
        - In the GUI shown in your browser, browse to the desired location (maybe a folder for this class?) and create a new Python 2 notebook..
        - At the prompt, type `print “Hello World!”` and press shift + enter (or select the “play” icon at the top, which has been pretty standard looking since the days of cassette tapes…). The string should be echoed below your command. (If you get an error about the kernel, try starting a command prompt (like above) and entering two commands: `conda install ipython` and, after that has run, `ipython kernel install`. Restart Jupyter and try again.)
    - Using Spyder.
        - Spyder works like any other IDE. This is probably where you want to do the assignments because you can save `.py` files, which is the required format for submission on CCLE. See the video playlist for more information.

  You are welcome to use any of the options above for the rest of these preparation exercises. I suggest switching between Jupyter and Spyder to get experience with both; each has its advantages.

- Review http://stephensugden.com/crash_into_python/Syntax.html
  - Note that Example 1 doesn't do anything meaningful. For instance, the statement `if True` is as silly in Python as it would be in any other language (the condition is always true, and the corresponding code will always run). This example is not a real program, it just shows you what the syntax looks like.
  - Don't worry if not everything looks familiar (`raise SystemExit`, for example) right now, but this should give you a basic idea of the difference between Python syntax and the language(s) you're used to.
  - Based on the example, try translating the following C++ program into Python:

```
// Example program
#include <iostream>
#include <string>
#include <cmath>
#include <string>
using namespace std;

int main()
{
  int N = 7;
  int i = 2;
  bool isPrime = true;
  while (i < N && isPrime == true){
      if (N % i == 0){
          isPrime = false;
      }
      i++;
  }

  string message;
  if (isPrime){
      message = "prime";
  } else {
     message = "not prime";
  }

  cout << message;
  return 0;

}
```

Here are a few hints:
Did the Python example need any include statements? Did it need a main function? No. Just start coding.
You don't need to declare variables by specifying the type. You just initialize variables.
The syntax of `while` and `if` constructs is different from C++/Java, but the logic is exactly the same.
Is `++` a valid operator in Python? (Nope.) How else can you do the same thing?
`&&` is not an operator in Python, either. The equivalent is the keyword `and`.
If you find this too difficult, don't break the computer. Move on and come back to it later.

- ☐ Read https://docs.python.org/2/tutorial/. The following readings are linked from there.
- ☐ Read “1. Whetting Your Appetite”.
- ☐ (Optional) Visit https://www.youtube.com/channel/UCGm3CO6LPcN-Y7HIuyE0Rew. Feel free to return when you need a break.
- ☐ Read the text just under 2.1 and in 2.1.2 (2.1.1 not required). A “tty” is a text-only console, like “terminal” on a Mac or the command window (cmd) in Windows. The main points here are how to start Python, how to get out, the use of line-editing features, and the difference between the primary and secondary prompts in interactive mode.
- ☐ Read the text just under 3. Note that in Spyder, you can select multiple lines of text and press ctrl + 1 to comment/uncomment all of them at once.
- ☐ Read all of 3.1 except 3.1.3, and read 3.2
  - Python “methods” are like member functions in C++ and non-static methods in Java.
  - Python “functions” are like non-member functions in C++ and static methods in Java.
  - “Slicing” may be new to you. Yes, it’s important.
- ☐ Practice with the following exercises:

Use Python to calculate:
Two to the power of five (should be 32)
The remainder of 3.5 divided by 1.1 (should be 0.2)
3.5 divided by 1.1 rounded down (should be 3.0)

Predict the result of the following operations:

```
3/5
3.0/5
3.0//5.0
```

Predict (in writing) and check the output of the following.

```
print "Frankie's"                       print "Frankie" + "\'s"

print "Frankie\'s"                      print "Frankie" "\'s"

print 'Frankie's'                       print """Frankie

print 'Frankie\'s'                               \'s"""

print r"Frankie\'s"                     print """Frankie \

print 3*"Frankie\'s"                             \'s"""
```

Predict (in writing) and check the output of the following, assuming that

```
s = "abcdefg"
```

has been executed first

```
print s[0]                              print s[:3]

print s[6]                              print s[3:]

print s[7]                              print s[0:-2]

print s[-1]                             print s[0:999]

print s[1:3]                            s[0] = 'z'; print s
```

```
s[0:3] = ['x','y','z']; print s         len(s)
```

Predict and check the output of the following, assuming that

```
s = ['a','b','c','d','e'] + ['f','g']
```

has been executed first. More importantly, which of the following will produce different output from the previous exercise, and what will it be?

```
print s[0]                              print s[3:]

print s[6]                              print s[0:-2]

print s[7]                              print s[0:999]

print s[-1]                             s[0] = 'z'; print s

print s[1:3]                            s[0:3] = ['x','y','z']; print s

print s[:3]                             len(s)
```