Computer Science 1 — CSCI 1100
Lab 0 — Getting Started
Fall Semester 2019

Lab Overview

The primary goal of this lab is get everyone started with the Python environment you will be using this semester, including everything from Python modules, to setting up a Dropbox, to starting to use the Spyder IDE. We also have an optional checkpoint for no credit that involves playing with an example Python program from Lecture 1. The program finds the one word in the English language that has three consecutive double letters. If you complete these goals, you will have done well in this first and perhaps chaotic lab.

Due to the vagaries of this semester's academic calendar, we need you to complete this lab prior to your first lab class on September 10 or 11. We apologize for needing to do this. We will, however, be holding an optionsl lab "help" session from 6:00-8:00pm on Thursday September 5th in DCC 308 and DCC 318. Please feel free to come there at 6:00 with any questions or issues and we will try to resolve them at that time. This is the only lab that you will be required to complete on your own outside of a formal lab period.

This lab will be due at the start of your first lab period on 9/10 or 9/11. If you are unable to finish this lab during the help sesson, do not worry, but please complete this lab afterwards and, if you need it, seek help from the TAs during their office hours (see `https://www.cs.rpi.edu/academics/courses/fall19/csci1100/index.html` — you can visit the office hours of **any** TA), you can also seek help from other students in the class, or from our forum on Submitty (https://submitty.cs.rpi.edu/f19/csci1100/forum/threads)! In future labs, we will only give you full credit if you complete the lab during lab time, but for this one, you will get full credit so long as you are ready before your personal lab 1 period.

Checkpoint 1: Setting Up The Environment

Please proceed through the following:

1. Make sure you can access the Submitty Discussion Forum by navigating to

   `https://submitty.cs.rpi.edu/f19/csci1100/forum/threads`

   .

2. Go to the course web site:

   `http://www.cs.rpi.edu/academics/courses/fall19/csci1100`

   and click on the **Software Installation** link for setting up the Python environment. This page contains the instructions for setting up the Python environment you will be using for this class.

3. Please proceed through the installation steps, all the way through getting the IDE and starting Dropbox. At the end, you should have your environment running, have tested the Spyder IDE, and have a Dropbox (or equivalent cloud storage) all set up. In short, you should be ready to go!

4. Now we are going to test your installation. Start by making sure you have created a folder for Lab 0 in your Dropbox. Then

   (a) Go to the **Course Materials** page on the Submitty website:
   `https://submitty.cs.rpi.edu/f19/csci1100/course_materials`

   (b) Click on the **Labs\Lab00** folder. You will see two python files there. The first is our friend `three_doubles.py` from Lecture 1 and the second is a short program `csci1100pythonmodulestest.py` we will use to get a sanity test on our installation. We will need both of these, so download them into your Lab 0 folder.

   (c) Start up the Spyder IDE.

   (d) Under the Spyder IDE File menu, load the `csci1100pythonmodulestest.py` file you just downloaded. You will see the code in the Spyder editor pane.

   (e) Look for the green arrow icon across the top of the IDE. Click on this to run the program. You will see it running in the pane on the lower right. This pane is the Python interpreter.

   (f) Two things should happen. First, a rotated version of the RPI logo will appear, and second, the interpreter pane will show something like:

```
[evaluate csci1100-python-modules-test.py]
Trying:
    test_doctest(1)
Expecting:
    1
ok
Trying:
    test_doctest(0)
Expecting:
    0
ok
2 items had no tests:
    __main__
    __main__.test_PIL
1 items passed all tests:
2 tests in __main__.test_doctest
2 tests in 3 items.
2 passed and 0 failed.
Test passed.
Testing PIL: RPI logo rotated 90 degrees is shown.
```

In order to complete this checkpoint, raise your hand and show a TA or a mentor that (a) you are able to access Submitty Discussion Forum, (b) you have the environment installed and tested, (c) you have a Dropbox set up, and (d) you loaded and ran `csci1100pythonmodulestest.py`. Note that if your test fails, it indicates that there is an issue with either the **doctest** or **PIL** installations. You can still get credit for the checkpoint if there is an issue, but you will need them later in the semester. Please come to office hours so that we can get your installation corrected as soon as possible! You will not be able to complete all of your assignments unless these tests can pass.

Checkpoint 2: Playing with the Triple Double Letter Code (Optional — no credit)

For this next checkpoint, we will play with the Python code we examined in Lecture 1. Proceed with the following steps:

1. Under the Spyder IDE File menu, load the `three_doubles.py` file you downloaded earlier. You will see the code in the Spyder IDE editor pane.

2. Look for the green arrow icon across the top of the IDE. Click on this to run the program. You will see it running in the pane on the lower right. This pane is the Python interpreter.

3. Now, see if you can figure out how to change the Python code in `three_doubles.py` to count the following:

   (i) number of words that have at least two consecutive pairs of double letters,

   (ii) number of words that have three pairs of double letters, and one letter in between each pair (for example: suddenness),

   (iii) number of words that have three pairs of double letters, and two letters in between each pair.

   In each case, there are two types of changes to make. Try and see if you can figure at least one of these out. Try to get them all. But,

   (a) Don't worry if you can't figure it out. It should be fun to try, and ask a fellow student, a TA or a mentor if you can't!

   (b) If you make mistakes, see if you can distinguish between syntax errors (errors that keep you program from running) and semantic errors (errors that give wrong results). Since you are unfamiliar with the syntax of Python at this point, only make changes slowly and refer back to the original version if you run into too much trouble.