

BST281: Lab Session 1

Announcements:

- Welcome!
- Administrative business:
 1. Email: xuz943@mail.harvard.edu
 2. Office hours: Wed 12:30-1:30pm
 3. Xue's office: CLS 11021
 4. Lab session hour: Mon 5:30Pm - 7:20pm
 5. Lab session location: FXB G11 (only 2/6 at Kresge LL6)
- ***Changes in this week's TA office hour*** (1/31 Tue12:00-1:00pm at CLS 11021)
- We are happy to answer questions that you have, but unless your question involves specific code please use the discussion board!
- For emails please use the subject: BST281
- Homework is due at 11:55pm on Friday biweekly.
- Question about HW grads should be directed to me
- If you need an extension, be sure to let us know **early**. Extension will only be granted:
 - For a good reason. "I started my HW late" is not a good reason

Setting up Python

Installing Python

In order for the operating system to understand what the command python means, it must know how to locate that program from the command line. Here are the instructions to get everything up and running.

1. Go to <https://www.continuum.io/downloads> download the appropriate version of Python 3.5 for your operating system and install it. As of this writing, the following caveats hold:
 - a) Many 64-bit extensions don't work very well, so even if you're running 64-bit Windows, install 32-bit Python.
2. After Python has fully installed, add it to your path:
 - (a) On Windows:
 - i. Download either the 64-Bit installer or 32-Bit installer
 1. Download the installer
 2. Optional: Verify data integrity with [MD5 or SHA-256](#) [More info](#)
 3. Double-click the .exe file to install Anaconda and follow the instructions on the screen
 - (b) On MacOS:

- i. Download either the graphical installer or the command-line installer

Graphical Installer

1. Download the graphical installer
2. Double-click the downloaded **.pkg** file and follow the instructions

Command Line Installer

1. Download the command-line installer
2. Optional: Verify data integrity with [MD5](#) or [SHA-256](#) [More info](#)
3. In your terminal window type one of the below and follow the instructions:
Python 3.5 version

You should now be able to open a new terminal and type *python* and see something like:

```
Xues-MacBook-Pro:~ Scarlett$ python
Python 3.5.2 |Anaconda 4.2.0 (x86_64)| (default, Jul 2 2016, 17:52:12)
[GCC 4.2.1 Compatible Apple LLVM 4.2 (clang-425.0.28)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
```

This allows you to input python commands one at a time. You can quit by typing `quit()`. But because if we have something complicated to do this kind of programming is very tedious and inefficient, we generate text files with the appropriate Python commands and then ask Python to execute them. Next, we will install the editors which we will be using. You need only install your editor of choice.

Setting up Atom

Installing Atom

Atom is essentially a fancy text editor like Sublime Text or Notepad++. It allows syntax highlighting, line numbering, bracket matching, automatic indenting and other features that make it really useful for programming. It is also pretty intuitive, really easy to install on a Mac or Windows.

Youtube introductory video: https://www.youtube.com/watch?v=bo5MM2N_3tw

1. Download Atom: <https://atom.io/>
 - (a) On Windows:
 - i. Click: Download Windows Installer
 - ii. Unzip AtomSetup.exe and then install
 - (b) On MacOS:
 - i. Click: Download For Mac
 - ii. Unzip atom-mac.zip and then install

2. Let's try creating and running a test test.py file.
 - a) Open Atom; you'll see an untitled window.
 - b) Save the file as test.py somewhere on your machine, but remember the path to the file (the top dropdown bar).
 - c) We have now generated the test.jedit.py file. Add the following line to the bottom of the file:

```
print("Hello, world!")
```
 - d) Open up a terminal, and type **python test.py** followed by the path to your file. If all goes as planned, the string "Hello, world!" should appear as output. Congratulations!
3. Explore more functions of Atom
<https://flight-manual.atom.io/>

Atom Exploration

Install package in Atom

<http://flight-manual.atom.io/using-atom/sections/atom-packages/>

Atom debugging

<http://flight-manual.atom.io/hacking-atom/sections/debugging/>

1. Open terminal
2. cd path (where you installed Atom)

Other Atom Trick

<https://www.sitepoint.com/12-favorite-atom-tips-and-shortcuts-to-improve-your-workflow/>

Python Practice

Command line examples

The very simplest way to interact with Python is through the command line interpreter. Open python by typing the python command in the terminal and then run the following commands:

1.

```
>>> print("Nice to meet you")  
>>> name="Scarlett"
```
2.

```
>>> print("Nice to meet you %s" % name)
```
3. Try changing "name" to your own name in the above commands and see what happens
4.

```
>>> a=1  
>>> b=2  
>>> c=a+b
```
5.

```
>>> print("what is the sum of a+b? %d" %c)
```

Now that we have our editors installed, we can do some simple commands to see how everything works.

Open up the test.py file that you created in Atom. Enter the above three commands into the file, and then run the file. You should get the same output as when you entered in the commands through the interpreter.

Command line introduction

NOTE: Some very useful material related to this topic can be found in Haddock and Dunn, Chapter 4.

The first thing we want to do is start up the terminal so we can do some practice. Table 1 shows how to do this for Windows or Mac.

Windows	MacIntosh
Either select Run from the Start menu and type <code>cmd</code> , or select Start/Programs/Accessories/Command Prompt	Navigate to Applications/Utilities and launch Terminal

Table 1: How to start the terminal for Windows or Mac.

Commands for doing various common and useful tasks such as opening or running `_les`, changing directories, etc. are on the attached sheets at the back of the lab. The commands for performing these actions in Windows are in table 2, while the commands for performing these actions on a MacIntosh are in table 3. Some useful shortcuts are shown in table 4.

Practicing with the command line

To familiarize yourself with these commands, do the following:

1. Make an empty document of your choice (for example, a Word document or a text document) and save it in your home directory
2. Open the commandline.
3. From the commandline, navigate to your home directory. Confirm you are at the right place by listing the files in that directory and finding the document you just created.
4. Make a new directory called bst281 practice in your current directory.
5. Move your file into bst281 practice.
6. List all the files in bst281 practice.
7. Move into bst281 practice.
8. Remove your file

Basic command line navigation

See the **Table of Utilities**

Submitting a Python script

1. When you submit your homework, submit only one text file named **problems01.py**.
2. The very start of the start of the file should be the “shebang”: `#!/usr/bin/env python`

3. The file should start with a triple-quoted documentation string containing (in order):
 - Your name
 - The file name (eg. problems01.py)
 - Your course number (BST281) followed by a space and then the date
 - How many hours you spend on the assignment
 - A blank line
 - A paragraph describing your thoughts on the assignment, major problems you had, and any comments/suggestions
4. Only function definition should be outside the `__name__ == "__main__"` block. You define your functions outside that block but **run** them inside the block
5. **MAKE SURE IT RUNS!!!** Please keep your TA happy by testing your finished program executes properly: run `python problems01.py` on the command line. If you get an error, make sure to fix it before your submit your program.

Making a Python Script

1. Open a new file in the Atom, and save it as lab01_script.py
2. Put `#!/usr/bin/env python` as the first line of your file
3. Make the next four lines a docstring:

```
"""
<your name>
<today's date>
"""
```

4. See my example in Atom
5. Go to command line and run your program (python <path-to-lab01_script.py>). You should see the output "Hello, Word!"
6. Practice what you have learned from this lab!

Difference between python 2 and python 3

http://sebastianraschka.com/Articles/2014_python_2_3_key_diff.html