Lab 1

Computer Science 111
Spring 2014

Welcome to CS111! In this lab, you will learn how to use IDLE, how to write a simple Python program, and how to submit your assignments.

You need to know your AC userid and password to do this lab. If you don't know them, run upstairs to the IT department and ask (they'll want to see a photo ID).

1 Logging in

Sit at any machine in the lab. Press three keys, CTRL, ALT, and DELETE, all at the same time, to get a login screen. Login with your Amherst College username and password. (If you aren't an Amherst College student and haven't already been in touch with our IT department about getting accounts on our system, please talk to a professor right away.)

Once you're logged in, click on the icon at the bottom left and search for the program IDLE. Choose **IDLE 3.3.2** from the search list. A window labeled "Python 3.3.2 Shell" should pop up. This is called the **shell window**.

1.1 Writing interpreted programs.

One way to write Python programs to type statements one-by-one into the shell window.

Try the following examples (without the `>>>` prompt that IDLE supplies.) The `#` marks are for **comments** intended for a human reader such as yourself. You can type them but don't have to: IDLE ignores anything after the `#`, so it doesn't make a difference to the program.

```
>>> print(3+4)
>>> n = 3+4               #  n gets the value three plus four
>>> print(n)
>>> song = "spam"           #  song gets the string "spam"
>>> print (song, song, song, song,  song+"ity", song)
```

What just happened:

1. The first statement adds 3 and 4 and prints the result.
2. The second statement adds 3 and 4, and puts the result into a variable named `n`. Notice that, unline in math, the `=` is pronounced "gets." It denotes a transfer of a value from the right side to the left side of the equal sign.
3. The third statement prints the value of `n`.

4. In the fourth statement, the variable named `song` gets the string value "spam." A *string* is a collection of characters delimited by quotation marks, as opposed to, say, an integer like 7.

5. The fifth statement prints the value of `song` six times; the fifth one has "ity" glued to it.

6. Notice that in the first statement the `+` means "add two numbers" but in the last staement the `+` means "glue two strings together." We call this latter operation *string concatenation.*

Question: Do you see the difference between a **variable name** (`n` and `song`), a **variable value** (7 and "spam") and a **variable type** (integer and string)? Any questions? Now is a good time to ask.

Getting input from the user. Now try these statements. They invite input from the "user," the hypothetical person who will someday use your program. For now you have to play the role of user and type in some answers.

```
>>> quest  = input ("What is your quest? ")
>>> vel  =  int (input ("What is the velocity of an unladen swallow? " ))
>>> print ("No. The correct quest is: Not  ", quest)
>>> print ("No. The correct velocity is ",  5*vel)
```

1. The first statement prompts the user with the string shown, then reads in a string (everything the user types up to <RETURN>).

2. The second statement prompts the user, then reads the response. The response is then converted to an `integer` so it can be treated as a number in the program. If you (the user) don't type a valid number, you'll see an error message. Try it!

3. The print statements print strings and values, separated by commas. The asterisk `*` signifies multiplication.

2 Creating a directory for your CS 111 work

You'll need to create and save lots of Python programs this semester. If you are doing this lab on your own computer, make a folder called cs111 and plan to store your programs in it.

If you are doing this lab on an AC computer, do the following. Double-click on the Computer icon at the top left corner of the screen. Double-click on the U: drive, and then choose "New folder". Call the folder "cs111". Go into cs111 and make another folder, "lab1". That's where you'll save today's work.

Note: If you leave your files on the Desktop of an AC machine they won't stay, because IC wipes desktops clean overnight. That's why you need to save it on your U: Drive.

3 Creating a `.py` file

You can create a program, save it, and reuse it as follows.

Within the shell window, choose the **File** menu item and then select **New Window**. Type the following program into it line by line:

```
tax_rate = 0.0625                    # massachusetts tax rate
price = int (input ("How much does it cost?" ))
total =  price + price * tax_rate
print ("That will be $", total)
print ("Thank you. Have a nice day.")
```

Now do this:

1. Choose **Save** from the File menu, and save the program into a file called **taxes.py**. (Be sure to save it in the cs111 directory that you created above.)

2. Go to the **Run** menu and choose **Run Module**.

3. If your syntax is exactly perfect, your shell window should reappear and your program will run, showing you the "How much" prompt. Try it!

4. If you see an error message, it is probably because have a typo in the program, called a *syntax error*. Programming languages generally require perfect syntax, including spelling and punctuation exactly as shown. Go back to the File window and fix your syntax errors. Then save it and run it again.

5. Or maybe the program crashed because the user (you) didn't provide a number. We will see how to fix this kind of error eventually, but for now just try to be a better user!

This program could be improved a little. Try changing it as suggested below.

1. If the user types a decimal number (dollars and cents) the program crashes because integers can't have decimal points. Change **int** to **float** on the input line, and re-run it. Now it will accept, for example, 45.99. (The `float` type is for "floating point decimal numbers," since you ask.)

2. Now that you've changed it to a **float**, the total has too many decimal places. To print a dollar sign and the total with two decimal places, change the `print` statement as shown below, and then run it again.

   ```
   print ("That will be", "$" + format ( total, "0.2f") )
   ```

   The plus sign is for attaching two strings (the dollar sign and the format string). The format thingy says, "make a string, treating the value in total as a floating point number with 2 decimal places."

4 Your assignment

Each week's lab has an assignment for you to demonstrate that you did the work. If you can't finish the assignment by the end of lab time, that's ok. Just finish it sometime over the weekend and turn it in by Monday morning. Lab assignments will normally get a check for being turned in, and you will be sent comments if there is something worth mentioning.

Week 1 Assignment

- Your assignment is to write a Python program that prints out an original haiku describing your feelings about computers and/or Python. A haiku is a 3-line poem: Typically in Engilsh the first and third lines have 5 syllables and the second line has 7 syllables. It should have a "cut," a juxtaposition of two images or ideas (like near vs far, or sound vs sight) with a *kireji* (cutting word) between them. The poem does not need to rhyme.
- At the top of the program, in a comment line, write your name. It is good style and convenient for the professor if you put your name inside the program.
- Also in a comment, indicate whether you want your poem to be entered in the *Spring 2014 CS111 Computer Poetry Competition*. Exciting prizes will be awarded!

5 Submitting your work

You can submit programs electronically:

1. First save your Python program. Call it **mypoem.py**.
2. Use a browser to visit `http://www.cs.amherst.edu/Submit`
3. Login, choose CS111 from the list of classes, and then choose "Lab 1". This will take you to the submission page for today's assignment.
4. Click the "Browse" button, and choose your "lab1.py" file. Click "Upload" and the file will be sent to us.
5. As the semester proceeds, you may find that you want to change your program after you submit it. That's fine. You can submit each assignment multiple times if necessary. We'll get the different versions and will know when each was submitted.
6. You can review your submission after you've submitted it by clicking on the appropriate link on the submission page.

6 Finishing up

When you're done working, select Quit IDLE in the IDLE menu, and exit your browser. Then click on the icon in the lower left corner of the screen and choose "Log off". Congratulations! Your first lab is finished!