Scripting for Data Analysis
Thursday, January 19, 2017

Python Language Basics: Values, Variables, Types, Expressions, Statements, Comments and Programs

In this class, we start a quick review of Python fundamentals. This should be a review of Python for undergrads who have taken IST 256, and an introduction to Python for graduates who have already learned another programming language such as R.

Python for Informatics from Charles Severance: http://www.pythonlearn.com/
The Python 3 version in html format: https://books.trinket.io/pfe/index.html
For this week, we are using a small part of Chapter 1 and all of Chapter 2

Python Interpreter

Assuming that you have Python 3.x installed, on Windows open a command prompt window or on a Mac open a terminal window, and type python

% python

and you should get some messages about your version of Python followed by the Python interpreter prompt:

>>>

From this prompt, you can type in little bits of Python and it will evaluate them and print the result. First we put in some values that are numbers and strings, where strings may have single or double quotes.

```
>>> 7
>>> 7 + 2
>>> 'Hello World'
>>> "Don't do that!"
```

Later, we will see how to type more complex values like lists.

Variables, Types and Expressions

When you just type a value, then it is not saved anywhere. To save a value, we put that value into a variable, by using assignment. For assignment, you put a variable name on the left, an equal sign, and an expression on the right. The expression is evaluated and the value is assigned to the variable. After you have assigned a value to a variable, it will keep that value (for printing or in expressions) until the end of the program or function, unless it is assigned another value.

```
>>> number = 7
>>> number
>>> average = (number + 9) / 2
>>> average
```

In expressions using numbers, we can use +, -, * and / for the standard arithmetic operators of addition, subtraction, multiplication and division, with parentheses to indicate grouping. (A standard order of operations is described in the text.)

For strings, we can use the + operator for catenation, sometimes called concatenation.

```
>>> string1 = 'Monty'
>>> string2 = 'Python'
>>> string1 + string2
```

So far, we have referred to values as either being numbers or strings. But numbers can be integers or floating point (any number with a decimal point).

```
>>> number2 = 7.2
>>> number + number2
```

Unlike some languages, the explicitly typed languages, you don't have to tell Python what type a value is, and it just figures out the most reasonable value. But in Python, every value does have a type, and the types can matter.

In expressions, there are also a number of functions that can be used with values and perform some operation. To invoke a function, you give the function name and give the value that you want it to operate on, called the parameter, in parentheses. One of these useful functions in the type() function.

```
>>> type (number)
>>> type (number2)
>>> type (string1)
```

Suppose that you define a variable

```
>>> balance = '8.5'
>>> type(balance)
```

Now if you type things wrong, you'll get some sort of error, so let's try some things, including

```
>>> balance + 2
>>> string3 = 'hello
```

What variable names should you use? The best advice is to try to make them meaningful in terms of your program. (I called these test examples things like number2 and string1

because they were just meant to be examples of that type.) There are rules about them: you can use letters, numbers, and _ (underbar) as long as the variable doesn't start with a number. Also suppose that you try

>>> if = 'hello'
This gives an error because "if" is a keyword in Python, so you must not use these keywords. However, you should also not use the names of types, like "int", "float", "string", "list" and a whole slew of types that will come up later.

One reason is that the type names are also functions (and you shouldn't redefine them). These type functions try to convert whatever value parameter you give it to the equivalent thing of that type. For example, suppose that you want to convert a number of type float to a number of type int.

>>> int (7.2)
And you can convert strings containing numeric characters to numeric types
>>> index = '8'
>>> int (index)
>>> float (balance)
but then try
>>> int(balance)

In the interpreter, we have seen that it will automatically print out the value of whatever you type in, but there is also an explicit print() function.

>>> print (balance)
>>> print ("The balance is", balance)

When you write Python programs, then you must explicitly use the print function to print out values.

User Input

Python has a built-in function called input that allows the user to type something. You can save whatever they type into a variable as a string.

>>> username = input('What is your name? ')
>>> print ('Hello', username)

Add the year that they were born:
>>> year = input('What year were you born? ')
>>> print ('Birth Year:', year)

But suppose that we want to use the year to compute the age. The year variable is type string, so we have to convert our value to an int first.

```
>>> type(year)
>>> age = 2017 – int(year)
>>> print ('Age:', age)
```

Exercises:
Submit your solutions in a single text file under the Assignments Tab for Week 1, due by next Tuesday, Jan. 24.
These exercises are from the Python for Informatics book.

Exercise Chapter 1-7:
What will the following code print out?

```
x = 43
x = x + 1
print (x)
```

a) 43
b) 44
c) x + 1
d) Error because x = x+1 is not possible mathematically

Submit your answer.

Exercise 2-3:
Write a sequence of statements into the Python interpreter to prompt the user for hours and rate per hour, printing each one, and then to compute gross pay as (hours * rate). Your output lines should look something like:

```
Enter Hours: 35
Enter Rate:  2.75
Pay:  96.25
```

Don't worry about making sure that Pay has exactly two digits after the decimal point.

Submit your code and the output by doing a copy/paste from the Python interpreter.

Exercise 2-4:
Assume that we execute the following assignment statements:

```
width = 17
height = 12.0
```

For each of the following expressions, write the value of the expression and its type.
1. width / 2
2. width / 2.0

3. height / 3
4. 1 + 2 * 5

Use the Python interpretor to check your asnwers. Submit your answers.

Please put your answers into one text file for submission.

And don't forget the blog reading from Tuesday.