Teaching Python

By: Sangho Bak

Simple Python Syntax:

Variables, in python are implicit and do not need to be declared. For example,

```
monty = True

python = 1.234

monty_python = python**2
```

Data Types, in python have some similarities and differences between the data types, for the scope of this project, I have listed some that you will have to know, notice that you already know some of the types.

String: A string of information use “ ” or ‘ ’, use str(x) to convert x to a string

Int: A simple holding of a number, use int(x) to convert x to an int

Float: A float, is short for a floating point real value, this basically works as decimal for us, float(x) to convert x to a float

Boolean: A Boolean returns either True or False result. No applicable conversion.

Whitespaces, are not very important in C# or java but it is very important in python. Whitespace is how you determine where a function/method ends and a new line of code in the project begins.

Comments, are the same and serve the same purpose, except that they use #.

```
#This would be a comment.
```

Arithmetic operations, simple arithmetic operations are very similar to ones from C#,

Addition: +, (a + b)

Subtraction: -, (a - b)

Multiplication: *, (a * b)

Division: / , (a / b)

Exponents: ** , (a**b)

Modulus: % , (a%b), gets the remainder of the two number 5%2 = 1

Note: Python automatically does int division, so be careful. 5/2 will return 2 not 2.5. To solve this issue, either set one of the numbers to a float 5.0, or 2.0. Or you may use the conversion method listed above in Data Types, which explicitly assigns the type.

Conditionals and Functions

If Statements, if statements in Python work in the same fashion as C# following all the same conventions and logic. Only the syntax is a little different.

```
if (some condition):
    #Do what you want it to do
else:
    #Do the other thing
```

However the other thing that is pretty different is how else if statements are done in python. The logic stays the same but now the naming convention is a bit different.

```
if(some condition):
    #do something
elif(some other condition):
    #do something else
else:
    #do the final thing
```

Note: You don't need the ( ) in the conditional if the problem is simple enough.

```
if(a > b):
if a > b:
```

are both acceptable ways to write an if statement.

Functions, act kind of like methods in our python. You package lines of code together to complete a task, then you can call the function repeatedly. The syntax of a function in python is very similar logically speaking to creating a method in C#.

```
def  name():
```

```
        #do something

        #do something

    #new line of code, not within the function.
```

These are the most important parts to remember when creating a function.

def - This is a keyword that stands for defining a function, this is needed to let the computer know that you are creating a function.

name() - This part is just deciding on a name, you can pick anything for the name of your function, you can decide on something like tax().

You can also have the function accept some kind of variable by placing it within the ( ) of the function. A common example would be something like tax(bill).

: - Don't forget to put this at the end of your ( ) this kind of acts like the {} in a method for C#.

Also, very important that you indent your lines of code within a function, that's how python knows that the code belongs in the function. Python will automatically know when the function ends, when it sees a line of code not indented.

Loops

For Loops, this loop iterates through every instance of a variable in a Dataframe/list. For the scope of this project I don't think it will be necessary, but I put it here just in case anyone was curious.

```
for variable in list_name:

    #do something
```

The breakdown for this falls like this.

for – keyword lets the computer know you're doing a for loop.

variable - the variable type within the list, int, string, obj.

in – keyword, acts as for each thing in this list

list_name – the list you're reading from.

While Loop, this loop will probably be the most important one for this assignment. While loops occur while a condition is true, and the condition is continuously updated. This might sound confusing in the beginning so I will show an example.

```
count = 0
while(count < 10):
    print str(count)
    count = count + 1
```

Now you might notice some striking differences for this loop, the loop keeps going while the conditional statement within the loop is true. In the case of the example as long as count is less than ten the loop will run.

Also notice the count, in and out of the loop. The first thing to note is that we create the count variable and set it to zero outside the loop. We do this so that the count can keep track of the iterations without being reset or losing scope. Lastly all while loops must have some way of being able to break the loop or else it will go on continuously and break the code. In this case we have it so that count is incremented until it is no longer less than 10.

While/Else, is a continuation of while loops, but python has a really special feature where you're able to make while/else loops, where the else trigger, if the while is false. Here is an example.

```
count = 0
while count < 3:
  num = random.randint(1, 6)
  print num
  if num == 5:
    print "Sorry, you lose!"
    break
  count += 1
```

```
else:
    print "You win!"
```

Notice how the else will trigger when the count is greater than three. Also notice the extra added information of losing when the number rolled is a 5. You can also see the keyword break which forcibly ends the loop.