**How to write a program/function**

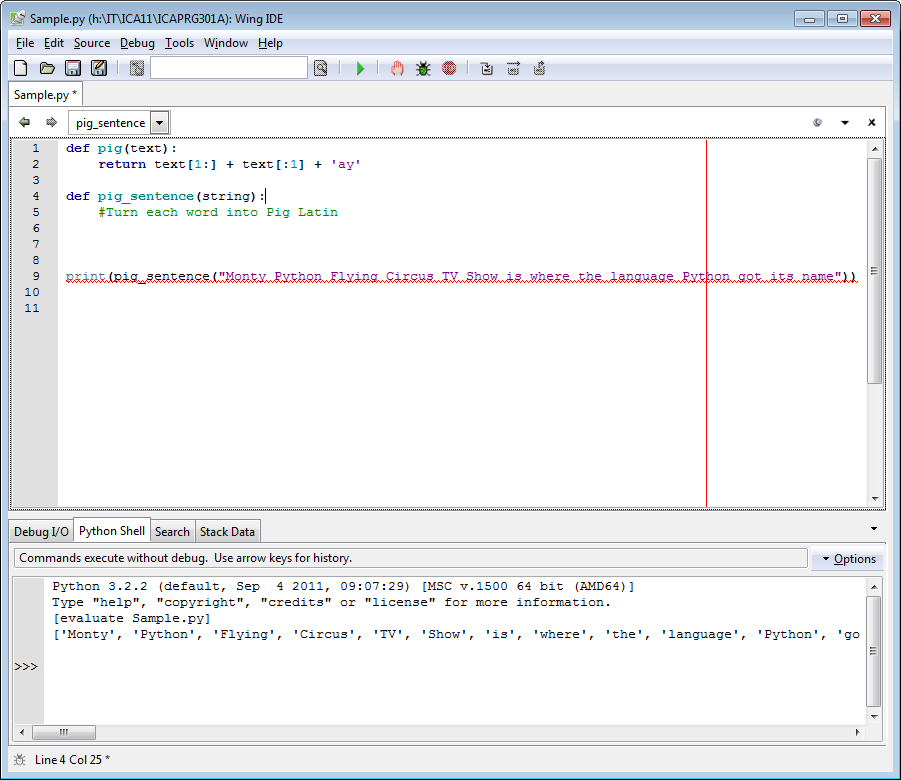
Most people when faced with a programming problem are faced with both a badly formed problem and a blank computer screen. For new programmer the first issue is WHERE DO I START?? There is a process that most programmers follow to turn this into a well formed program or function. We will go through these steps with the following exercise. The problem is:

*Create a function which turns a sentence into the sentence in Pig Latin.*

We already have a function for turning a word into Pig Latin (Lesson 2) so that is our starting point. We now need to clearly define what we need to do.

*We need to create a new function which will turn each word in the sentence into Pig Latin.*

Programming this in Python so far and we would have the following, I will use proper function names:



We had copied our Pig Latin function into the program and created a new function called pig\_sentence. Then down the bottom we will print out the answer for this. The red line shows that currently this will not work.

Inside the function pig\_sentence is a comment with what we think the function will do at this stage. However it is still not enough to actually code the answer so there is more thinking to do.

pig\_sentence is a function which takes a sentence (called String) and does something to it and gives back an answer. In programming (and especially in Python) it is not a good idea to change the original sentence (even if you could). The best solution is to create a new String and give that one back as the answer once we have built it properly. This is down using these two commands:

def pig\_sentence(string):

new\_string = ""

#Turn each word into Pig Latin

return new\_string

The first command creates a new string called new\_string that we set to a blank string. We are going to build this up to be the new string of Pig Latin based upon the old string. The final command (return new\_string) will send this answer back once we have finished.

In this function there will be two strings, one called string is the sentence that needs to be changed. The other called new\_string is the sentence once it has been changed. In Python, like most languages, Strings are immutable variables, they cannot be changed but they can be re-assigned. So in building the new\_string as the returned variable we keep the old sentence intact and unchanged.

Now let’s look at the comment. *Turn each word* is a clue that we need a loop which works on each word of the sentence while *into Pig Latin* tells us what we need to do to each word in the loop.

The next task is to turn the sentence (called string) into something that we can use to loop through. In Python you can loop using **for** or **while.** The one to use when moving through a defined set of data is the **for** loop. However this only works on list and what we have is a String. Now we need to think of how we can turn the sentence String into a list that can be looped through.

Here we can turn to Google for help. By searching using the keywords “python string to list” we get over 11 million results (last time I did it). The best answers come from Stack Overflow so let’s go there first. Here the simplest answer is using the String function split().

def pig\_sentence(string):

new\_string = ""

for word in string.split():

#Turn each word into Pig Latin

pig(word)

return new\_string

Now we can get to each word in the sentence (called word in the loop) and turn each word into its Pig Latin counterpart. The command pig(word) inside the loop will use our other function to produce the word in Pig Latin. It looks like we are finished, but this does not work.

We still have an issue, and it comes not defining our problem clearly enough. The issue is what we want is a new sentence. We have created it and returned it, but we still need to build it inside the loop. What we needed to say in our original problem was this:

*We need to create a new function which builds a new string by turning each word in the sentence into Pig Latin.*

So we need to build a string based upon the old string and the new information created in each run through the loop. This is a very common practise in programming and is one of the foundation stones for any programmer. The code for this is

new\_string = new\_string + " " + pig(word)

new\_string is a variable of the type String so it is immutable, it cannot be changed once it is created. However it can be re-assigned to a new value. This is exactly what this line does. In each run through the loop we add a space and then the pig latin version on the word to the previous version of the string.

Our function is now complete and it works. This is the process that programmers go through for every problem large or small. The steps are:

1. Redefine the problem into something that can be solved by code
2. Create an outline of code with comments where you want things done
3. Break the comments ups into section which can be coded
4. Work out the best way to do each section
5. Keep working through progressively adding code and redefining the problem until it is solved

The complete function is below:

def pig\_sentence(string):

new\_string = ""

for word in string.split():

#Turn each word into Pig Latin

new\_string = new\_string + " " + pig(word)

return new\_string