**Python 2: Getting More Out of Python  
Lesson 2, Project 1**

Handed in: 18 Feb 2015 02:48:37PM Graded: 18 Feb 2015 03:35:14PM

Here are your instructions:

Make a **UnitTesting\_Homework** project and assign it to the **Python2\_Homework** working set. In this project, write a unittest test program for the following function. (The test program makes unittest.TestCase assertions about the results of calling the function with known arguments.)

def title(s):

"How close is this function to str.title()?"

return s[0].upper()+s[1:]

Test your results for a given string s by comparing them with s.title(). Because this is purely an exercise, it's OK to put your test code in the same file as the function and just hand in a single file. Your file should be an importable module. You should be able to find an example that shows title(s) and s.title() diverge (have different output). Bonus marks for fixing the error in the function above (making it behave more like the native method).

**Your Comment:**

*no comment given*

**Items Handed In**

* [Open Project Handed In](file:///C:\Code\O'Reilly%20School\Python%202\Lesson%202%20-%20Unit%20Testing\project\%3f\.handin\160-7828-1\com.ost.jwoloson.160.7828.1.UnitTesting_Homework.zip)

**Overall Comments:**

Hi Jason,

This is a solid 1st attempt. Turning this back to your request, but figured have a look at it. It is a little bit of feedback:

- It's a really good idea to keep your lines of (indentations + code + comments) to <79 characters. This is partly for readability - no horizontal scrolling required - and partly to comply with Pythons style guidelines.

If you're interested, you can pick up your own copy of "PEP-8" at:

<https://www.python.org/dev/peps/pep-0008>

... this can be a little bit tough when writing test methods, but you can usually break up lines like this:

def test\_title\_func(self):

'''Tests effectiveness of 'title' function'''

msg="This function perfectly matches the str.title() method"

self.assertEqual(title(s),

s.title(),

msg)

-Keep in mind the circumstance under which the fail message gets printed. It's only the print if your assertion is not correct. Therefore the message you currently have would be misleading.

Your tests look good, generally. In the future, you might consider building everything using variables. As an example, instead of doing something like this:

def test\_multiple\_words(self):

self.assertEqual(title("some string"), s.title("some string"), "the strings aren't the same")

... you could go something like:

def test\_multiple\_words(self):

s = "this is a test"

self.assertEqual(title(s), s.title(),

"\nTesting: {}\nThese should be the same: \nbuilt-in: {}\ntitle(): {}".

format(s, s.title(), title(s))

)

Note that the fail messages are verbose enough to be useful in quickly identifying and tracking down any issues. You'd see something like this:

AssertionError: These should be the same:

built-in: This Is A Te'St

title(): This Is A Te'st

Note also that the string being tested is a variable, and that its contents need be stated only once. If you do it this way, you have the possibility to reuse the test, or even wrap it into a loop. Something like this:

def test\_loopy\_words(self):

for s in ["this is a te'st",

"they're on to us!",

'test',

]:

self.assertEqual(title(s), s.title(),

"These should be the same: \nbuilt-in: {}\ntitle(): {}".

format(s.title(), title(s))

)

- The purposes of these tests is to beat the living \*#& out of your main app. In baseball terms, you want to throw hard balls, curves, sliders and maybe a spitball or 2 for good measure. You want it to break on your watch, not your clients'.

-As a point of potential interest, it's possible to work this out so that an instance of divergence can actually result in a passing test. Here's an example:

def test\_irish\_president(self):

"""

The 2008 election campaign was remarkable for the fact that the Democratic and Republican presidential and vice-presidential

candidates possessed elements of Irish ancestry. Much more remarkable of course was the fact that the victor was Barack Obama,

the United States's first African-American President. Obama is Kenyan in his paternal ancestry,

but his forebears on his maternal side include a great-great-great grandfather Falmouth or Fulmuth Kearney or Carney,

born in Ireland about 1832 and emigrated to the US in 1850, whose father is stated to be

Joseph Kearney, a shoemaker of Moneygall, County Offaly (<http://www.wargs.com/political/obama.html>).

US genealogist Megan Smolenyak first established Obama's Moneygall connection, and for further information on her researches

see <http://www.rootstelevision.com/blogs/megans-rootsworld>.

"""

s = "barack o'bama"

self.assertNotEqual(s.title(),

title(s),

"s.title should result in \"{}\"".

format(s.title())

)

-Pat

**Grade:**

Try Again

Your instructor would like you to work on this project. Be sure to hand in your corrections!