CS 1110, LAB 5: CONDITIONALS AND ASSERTS

http://www.cs.cornell.edu/courses/cs11102013fa/labs/lab05.pdf

First Name: ____________________ **Last Name**: ____________________ **NetID**: ______________

You have now had an extensive assignment that made you an expert of string slicing. However, none of the functions in that assignment required conditionals. This lab builds upon those skills, and gives you experience writing more complex functions involving conditionals.

In addition to conditionals, we have some questions reguarding asserts and error handling. The exercises in this lab are prime exam questions. In fact, the Pig Latin question is actually from Prelim 1 in a previous semester of CS 1110.

Lab Materials. We have created a single file for this lab: `piglatin.py`. You can download it from the Labs section of the course web page. This lab will require that you modify this file, as well as answer some questions on this worksheet. You need to show both to your instructor to get credit for the lab.

Getting Credit for the Lab. When you are done, show both this handout and `piglatin.py` to your instructor. You instructor will then swipe your ID card to record your success. You do not need to submit the paper with your answers, and you do not need to submit the computer file.

As with all previous labs, if you do not finish during the lab, you have **until the beginning of lab next week to finish it**. You should always do your best to finish during lab hours. Remember that labs are graded on effort, not correctness.

1. Pig Latin

Pig Latin is childish encoding of English that adheres to the following rules:

(1) The vowels are ‘a’, ‘e’, ‘i’, ‘o’, ‘u’. A ‘y’ that is *not* the first letter of a word is also considered a vowel. All other letters are consonants. For example, “yearly” has three vowels (‘e’, ‘a’, and the last ‘y’) and three consonants (the first ‘y’, ‘r’, and ‘l’).

(2) If the English word begins with a vowel, append “hay” to the end of the word to get the Pig Latin equivalent. For example, “ask” becomes “askhay,” “use” becomes “usehay.”

(3) If the English word starts with ‘q’, assume it is followed by ‘u’; move “qu” to the end of the word, and append “ay”. Hence “quiet” becomes “ietquay,” “quay” becomes “ayquay.”

(4) If the English word begins with a consonant, all the consonant letters up to the first vowel (if any) are moved to the end of the word, and “ay” is appended to get the Pig Latin equivalent. For example, “tomato” becomes “omatotay,” “school” becomes “oolschay,” “you” becomes “ouyay,” “my” becomes “ymay,” and “ssssh” becomes ”sssshay.”

Course authors: D. Gries, L. Lee, S. Marschner, W. White

Your goal is to write a function pigify that take takes a single English word (e.g. a string with only letters and no spaces), and converts it into Pig Latin.

1.1. **The Function first_vowel().** To aid with our Pig Latin conversion, we have provided a helper function first_vowel(w), with the following specification:

```
def first_vowel(w):
    """Returns: position of the first vowel; -1 if no vowels.

    Precondition: w is a nonempty string with only lowercase letters"""
```

We hope that this helper function is correct. To verify this, write down at least 5 key test cases (each should be testing a different "idea", so do not just say 'a', 'e', 'i', 'o', and 'u'). **We do not want you to write a test script or try to fix any bugs**. Just write down the test cases.

1.2. **The Function pigify.** The function pigify() has a short-and-simple specification:

```
def pigify(w):
    """Returns: copy of w converted to Pig Latin.

    Precondition: w is a nonempty string with only lowercase letters"""
```

This specification assumes that you have read the definition of Pig Latin on the previous page. Implement this function in piglatin.py When you are done, you will want to test your answer. Instead of creating a unit test, we only want you to write down a list of test cases to verify that your implementation is correct.

2. Asserts

Strings have the following useful methods:

Method	Description
s.isalpha()	**Returns**: True if `s` is nonempty and has only letters; False otherwise.
s.isdigit()	**Returns**: True if `s` is nonempty and has only numbers; False otherwise.
s.islower()	**Returns**: True if all letters in `s` are lower case; False otherwise.
s.isupper()	**Returns**: True if all letters in `s` are upper case; False otherwise.

Now go back and read the precondition for the function `pigify`. In the box below, write assert statement(s) that enforce this precondition. Remember to check the type as well!

Add these asserts to `pigify` in `piglatin.py`. Now suppose you had the following function (you can add this to `piglatin.py` if you wish):

```
def try_pigify(w)
    """Returns: copy of w converted to Pig Latin.
    Returns The empty string if w is not a lower case string of letters only.

    Precondition: there is no precondition"""
    try:
        return pigify(w)

    except:
        return ''
```

Suppose you executed the assignment statement `s = try_pigify('Piggy')`. What is now in the variable `s`; **explain why**.

On the other hand, suppose you executed the statement `s = try_pigify('piggy')`. Now what is in the variable `s` and why?

When you have answered this question, show your work to your instructor. You are done.