CS 1110, LAB 4: STRINGS AND CONDITIONALS

Name: _____ **Net-ID**: _____

There is an online version of these instructions at

http://www.cs.cornell.edu/courses/cs1110/2012fa/labs/lab4

You may wish to use that version of the instructions.

You have already 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 the skills from the first assignment, and gives you experience writing more complex functions involving conditionals.

We recognize that this is a busy week with the first assignment revisions and the second (written) mini-assignment. Therefore, we have tried to keep this lab as short as possible so that you can focus on your other work. The activity in this lab has actually been a Prelim 1 question in past semesters of CS 1110, though this assumes more experience with conditionals that you have yet had.

Requirements For This Lab. The very first thing that you should do in this lab is to download the file `piglatin.py` from the course web page:

http://www.cs.cornell.edu/courses/cs1110/2012fa/labs/lab4/piglatin.py

This is a module with two functions. One function is fully implemented, but you are expected to come up with test cases for it. The second function is a stub; this is the one that you are to implement.

There are only two things to show for this lab. First, you need to complete the function `pigify` in `piglatin.py` and show it to your instructor. **Do not turn in this module**; the instructor will look at it and record that you did it.

In addition, this worksheet asks you to write down several test cases. While you may wish to use these test cases to test your function, we are not requiring that you implement a unit test. Simply write the test cases on a piece of paper and show them to your instructor. When you demo your function `pigify` to the instructor, she or he may ask you run the function on a few of these test cases in the interactive shell.

As always, if you do not finish during the lab, you have **until the beginning of lab next week to finish it**. We have tried to make this lab as short as possible, given the long lab last week and the assignment. Additionally, remember that labs are graded on effort, not correctness.

On the other hand, if you finish early, then you are free to talk to the instructor and consultants about any questions that you have in the course.

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."

Our 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.

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.

    There is a better way to do this function with for-loops, but we have not covered
    that topic yet.

    Precondition:  w is a 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').

The Function `pigify`. The function `pigify(w)` has the following short-and-simple specification:

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

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

This specification assumes that you have read the definition of Pig Latin above. Implement this function in `piglatin.py`

When you are done, you will want to test your answer. Instead of creating a unit test, write down a list of test cases to verify that your implementation is correct.

Now you can show the module `piglatin.py` to your instructor. When you are demonstrating to the instructor, run the interactive shell and try out the function on a few of the test cases listed above.