CS 30 Lab 6 — Functions and Control Structures

This lab gives you more practice with functions, decisions, and loops. You are encouraged to work with a partner, and to talk things over with your lab-mates. Don't hesitate to call me or Dan over for help or answers to questions.

---

1. Pig Latin is an invented language formed by transforming each English word according to the following simple rules:

   - If the word begins with a consonant, you form its Pig Latin equivalent by moving the initial consonant string (that is, all the letters up to the first vowel) from the beginning of the word to the end and then adding the suffix *ay*.
   - If the word begins with a vowel, you just add the suffix *way*.
   - If the word contains no vowels, just add *ay*.

   For example, the word *scram* becomes *amscray*, and *apple* becomes *appleway*. Otgay itway? Oodgay! So let's write a program that repeatedly asks the user for a phrase, and prints out the corresponding Pig Latin translation. Here is the skeleton program from class:

```
def piglatin():
   while True:
      phrase = raw_input("Input: ")
      if phrase in ['bye', 'goodbye', 'quit']:
         print "So long!"
         break
      print translate(phrase)

def translate(phrase):
   return phrase
```

   Type this in and test it out in Python. You'll see that it repeatedly asks for a phrase until you enter "bye", "goodbye", or "quit".

2. Next, define a separate function called `firstVowelPos` which takes a single word as input and returns the numerical position of the first vowel (*a*, *e*, *i*, *o*, or *u*) in the word, or else -1 if the word contains no vowels. Be sure to test your function. For example:

   `firstVowelPos("apple")` should return `0`
   `firstVowelPos("cherry")` should return `2`
   `firstVowelPos("rhythm")` should return `-1`

3. Now use `firstVowelPos` in a new function called `translateWord`, which takes a single word as input and returns the Pig Latin equivalent. Remember that if *w* is a word, then the Python expression *w*[*m*:*n*] will return the letters from position *m* up through but not including position *n*. Be sure to test `translateWord`. Examples:

   `translateWord("scram")` should return `'amscray'`
   `translateWord("apple")` should return `'appleway'`
   `translateWord("rhythm")` should return `'rhythmay'`

(continued on back)

4. Finally, using `translateWord`, complete the definition of `translate` so that it splits the phrase into words, calls `translateWord` on each individual word, and assembles the results into the final Pig Latin phrase. Test your top-level `piglatin` function to make sure everything works.

   For an extra challenge, see if you can get your program to handle phrases containing punctuation and capitalization correctly (warning: this can be tricky).

5. A person's body mass index (BMI) is calculated as their weight (in pounds) times 720, divided by the square of their height (in inches). A BMI in the range 19-25, inclusive, is considered healthy. Write a program that calculates a person's BMI and prints a message telling them whether they are in the healthy range, or else need to see their doctor. Hint: in Python, conditional tests with multiple operators are permitted, such as `0 < x < 5`, in addition to single-operator tests like `x >= 3`.

6. Now that we have if-else and while statements in our toolbox, we can revisit the task of making a graphical image move around the screen indefinitely. Do exercise 17 on page 230 of the textbook, but instead of using a counted loop (*i.e.* a for loop), use a while loop that loops indefinitely.

7. The game of Rock-Paper-Scissors is played by two opponents, and consists of a series of rounds. On each round, the players simultaneously say either "rock", "paper", or "scissors", and the winner of the round is determined as follows:

   - If Rock and Scissors are chosen, Rock wins because Rock dulls Scissors.
   - If Paper and Rock are chosen, Paper wins because Paper covers Rock.
   - If Scissors and Paper are chosen, Scissors wins because Scissors cut Paper.
   - If the choices are the same, no one wins.

   Write a program to simulate a game of Rock-Paper-Scissors between you and the computer. Call your program `rps()`. On each round, the computer should make a random choice and then ask you for your choice, after which it should reveal its choice and report the winner of the round. The game continues until the user enters a blank line. Your program should behave as shown below (the user's input is in boldface). Note that invalid choices by the user are rejected.

```
>>> rps()
Welcome to Rock-Paper-Scissors!

Rock, Paper, or Scissors? paper
You chose paper, I chose scissors
Scissors cut paper, so I win!  Ha ha!

Rock, Paper, or Scissors? rock
You chose rock, I chose scissors
Rock dulls scissors, so you win.  Hmmmph.

Rock, Paper, or Scissors? rattlesnake
Hey, that's not a valid response!

Rock, Paper, or Scissors? scissors
We both chose scissors, so nobody wins.
```