Python Assignment Alternative #2

1. Read Sections 2.1 - 2.3 and 3.1 - 3.3 of the lecture notes posted online:

http://anh.cs.luc.edu/python/hands-on/3.0/handson.pdf

2. Complete the following programs

Part A: Finding the Words in a Sentence (findwords.py)

Write a python program that asks the user to enter a sentence and prints out each separate word in that sentence. You are guaranteed that the only characters except for letters in the sentence will be the following: space, tab, comma, period, question mark, excalamation point, semicolon, colon, single quote and double quote.

Sample Program Run (User input in bold)

```
What is your sentence?
Johnny said to Anna, "Where are you going?"
The words in your sentence are
Johnny said to Anna Where are you going
```

Part B: Radio Censor (radio.py)

Some words are not to be spoken on the radio. In the past, radio censors would either hang up on a caller who spoke these words, or bleep them out. Your job is to come up with a more sophisticated technology. You will be given a list of censored words and equivalent words that should be said in their place that are not censored. For example, "mean" might be censored, but its equivalent, "unpleasant" might not be. In this situation, your job would be to read in a message and change each censored word to its uncensored equivalent. You will first read in from the user the pairs of censored and matching uncensored words, and then read in a sentence to translate, according to these rules.

Sample Program Run (User input in bold)

```
How many censored words are there?
3
Please enter censored/uncensored pair #1:
mean unpleasant
Please enter censored/uncensored pair #2:
fat bulky
Please enter censored/uncensored pair #3:
obnoxious confident
Please enter a sentence with lowercase letters only:
the mean obnoxious bully yelled at fat people
```

```
The censored sentence is as follows:
the unpleasant confident bully yelld at bulky people
```

Part C: Radio Censor Part II - File Input (radio-file.py)

This time, rewrite the Radio Censor program to take input from a file. In particular, the input file format will be as follows:

The first line will contain a single positive integer, *n*, representing the number of censored words. The next *n* lines will contain two words each, the first word being the censored word and the second word being its replacement.

The following line will contain a single positive integer, *len*, which is the number of lines in the message to translate.

Your output should just be the entire message (line by line) translated with the censored words substituted. The output should go to a file.

Sample Program Run (User input in bold)

```
What is the input file?
message.txt
What file do you want to store your output?
safemessage.txt
Your message has been stored!
```

Sample Contents of message.txt

```
5
mean unpleasant
fat bulky
obnoxious confident
filthy unkempt
lazy unmotivated
3
the unmotivated filthy slob ate lots of fat every day
unfortunately this made him obnoxious and mean
luckily he was not lazy
```

Sample Contents of safemessage.txt

```
the unmotivated unkempt slob ate lots of bulky every day
unfortunately this made him confident and unpleasant
luckily he was not unmotivated
```

Part D: Wedding Invitations (invite.py)

Rewrite program 2A (Wedding Invitations) that you write in C in python. The output format, etc. of your python program should be identical to the C program you originally wrote.

Part E: Grid Search (search.py)

Write a program that originally generates a random spot on a x-y grid with minimum coordinates (0,0) and maximum coordinates (9,9). The bottom left corner of the grid will be (0, 0). Then, prompt the user for a random guess as to the secret grid location. If the user gets it on the nose, print out a congratulatory message. Otherwise, tell them which direction they must move in to get to the point. We will consider moving in the positive x direction as going East and moving in the positive y direction as going North. (Just use N, S, E and W. Sometimes you will print out two directions for movement, other times only one.) At the end, also print how many guesses they took.

Sample Run (User input in bold and italics)

```
Enter your guess for x and y.
3 5
This is not correct. You need to move NW.
Enter your guess for x and y.
1 7
This is not correct. you need to move S.
Enter your guess for x and y.
1 6
You got the secret location (1, 6) in 3 guesses!
```

Part F: Turtle (myturtle.py)

Though this isn't in the tutorial, using the turtle is simple enough, you should be able to pick it up with these few hints. First, in order to use the turtle, do the following import first:

```
import turtle
```

To make commands that the turtle listens to, you need to call each method (function) as follows:

```
turtle.methodname()
```

This turtle emulates the language Logo. In particular, the turtle has a pen. If the pen is down and the turtle moves forward, then a line is drawn. The turtle can turn right or left any number of degrees, move forward, lift its pen up or put it down, and it can draw a circle.

Also, it may be useful to use some functions in the math library when using the turtle. To use the math library do the following import:

```
import math
```

To call methods in the math library, you must put "math." (without the double quotes) before the method call. Here is a call to the math acos function:

```
angle = math.acos(1/2)
```

Note that acos returns an angle in radians, and by default, the turtle turns right or left in degrees. But, if we want, we can allow the turtle to take in angles by radians. We just have to run the following command:

```
turtle.radians
```

Here are some of the turtle functions:

```
forward(int x) - moves the turtle forward x pixels

right(double angle) - turns the turtle angle degrees (or radians) to the right

left(double angle) - turns the turtle angle degrees (or radians) to the left

reset() - clears the drawing screen and moves the turtle back to the middle, pointing to the right.

up() - picks up the pen so that in a subsequent command, nothing is drawn

down() - puts down the pen so that in a subsequent command moving forward, something is drawn.

circle(int radius, double angle) - draws a circle with the given radius for angle number of degrees (or radians). For example, if angle is 90 and the turtle is in degree mode, then a quarter of a circle is drawn.

color(String c) - sets the color of the pen to c. (Note: I am note sure which colors are supported via their names. So far, I have tried red, green and purple and they all work.)
```

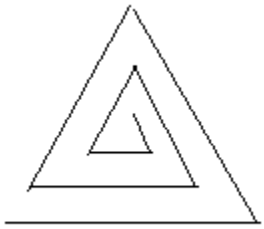
For a full description of what the turtle can do, go here:

http://docs.python.org/library/turtle.html

For this part of the assignment, write two functions:

1) A function that writes out your name, tracing out each letter using the turtle. You may make your name in any color you wish. Feel free to write more functions to aid you in this cause.

2) A function that prints out a spiraling triangle, similar to the one shown below:



Your function should take in length of the maximum side in the triangle and draw a corresponding triangle where each side length is 10 pixels more than the previous side drawn. (Technically, in this drawing, the last side should be extended by 10 pixels. To make the figure look better, you can make the last two sides the same length, if you like as in the figure above.)

In your main function, just call both of these functions so both your name and an example spiral triangle print out. You may choose where both items print on the screen.

Note: It's okay if your triangle prints in a different orientation with the horizontal side being on the top instead of the bottom.