Lab 4: The Vigenère Cipher

This lab is intended to give you some practice using strings and loops.

How the Cipher Works

Take a look at the Wikipedia entry for all the details on the Vigenère cipher:

http://en.wikipedia.org/wiki/Vigenre_cipher

Here's the executive summary. The cipher uses a word for a key, such as "batleth". The key is treated as a series of numbers, like this:

```
b a t  l  e t  h
2 1 20 12 5 20 8
```

To encrypt a phrase, each letter is shifted by the amount indicated by the corresponding number in the key. A demonstration is the best way to explain this:

```
Three Galor-class vessels are heading toward your position!
```

First, the phrase is usually stripped of spaces and punctuation. This is optional - if you like, you can leave them in. For this demonstration, I'll take them out and convert the entire phrase to upper-case:

```
THREEGALORCLASSVESSELSAREHEADINGTOWARDYOURPOSITION
```

Next, each character is shifted as shown (with the shift as a letter - the numerical shifts corresponding with each letter are listed above). The alphabet wraps around, so the letter after Z is A.

```
Character: THREEGALORCLASSVESSELSAREHEADINGTOWARDYOURPOSITION
Shift    : batlethbatlethbatlethbatlethbatlethbatlethbatlethb
Result   : VILQJAINPLOQUAUWYEXYTUBLQMYIFJHSYIECSXKTOZRPMUYCWP
```

Now the text is encrypted. There are two methods by which you can decrypt it:

1. Shift each letter back the appropriate number of spaces (just like encryption, but shift backwards instead of forwards)
2. Calculate a decryption key which will shift each letter enough to get it back to where it started. So if the letter was shifted 2 places during the encryption sequence, it should be shifted 24 places for decryption. The matching key to "batleth" is "xyfnufr", which is slightly harder to remember. Here's a bit of Python code which will calculate a decryption key (note that the resulting key will be upper case):

   ```
   dec_key = ""
   for c in key.upper():
       dec_key += chr(90 - (ord(c) - 64)%26)
   ```

Writing a program to do this

I'd recommend starting with the shift cipher example from class ("shift.py", posted with the other in-class examples). You'll need to make a number of modifications:

- Instead of entering a number for the shift, the user will enter a word as the key
- You must keep track of which letter of the key you are on, as well as which letter of the plaintext. I would suggest something like this (the following assumes your key is named `key`):

  ```
  key_spot = 0
  for c in plaintext:
      # Use key[key_spot] to shift the current character
      key_spot += 1
      if key_spot >= len(key):
          key_spot = 0
  ```

  In other words, increase `key_spot` each time you encrypt a character, and when you get to the end of the key, start back at the beginning.
- The shifting will be done much differently. Change "shift" to the shift specified by the current letter of the key. Assuming that your key is named `key` and that it is all in uppercase (use the .upper() method if it is not), the following will generate the correct shift:

  ```
  shift = ord(key[key_spot] - 64)
  ```

- Decrypting will be more difficult. 8 of the 10 points will be awarded if the lab is perfect except for this part. You should either ask the user if the text is to be encrypted or decrypted, make a copy of your program which decrypts instead of encrypts, or generate a decryption key and print it out for the user (see above).

Making your program more useful

It's not that hard to read the text from a file. Take a look here:
http://docs.python.org/library/stdtypes.html
This is not required, but will make your program more useful.