Programming Assignment 3

Getting started

This assignment is designed as a practice on programming with static methods, loops, conditionals and string operations.

Programming Project: Decoder **worth 20 points**

Decipher a message.

Your program will decode a message entered by the user by extracting the words between # symbols, reversing the letters in those words, and joining them together into a sentence. It must define and use three static methods specified below. Be sure to use the exact same method names and the same inputs/outputs as described.

1. Method **revLetters()** (4 points). This method must be passed one parameter of type String and return the reverse of that string. For example, when passed `"nose"`, it should return string: `"eson".`

   To test it, add code to the main method that invokes this method, passing it a string. Either add code to the main method that prints the returned string. While you will need to remove the code you add to the main method after testing, this kind of verification will save time in the long run.

2. Method **decrypt()** (9 points).

   This method must be passed one parameter of type String, extracts the letters between # symbols, invoking the revLetters() method to reverse the order of those letters, and then joins all the decrypted words together and returns the decrypted string.

   For example, if this method is passed the following input:

   `"the#woh#isinthe#era#mycat#uoy##gnileef#esuom#?yadot#so"`

   Then it must:

   - Create a variable that will contain the decrypted message and initialize it to an empty string.
   - Repeat the following steps starting from the beginning of the parameter string, until the whole parameter string has been processed:
     - extract the word between **the next two** # symbols (`"woh",` then `"era"),`
     - invoke **revLetters**() to reverse the letters (`"how",` then `"are")` ,
     - update the decrypted message by adding the extracted word to it (resulting in `"how",` then `"how are",` and so on`)`
   - Return the decrypted message `("how are you feeling today?")`

   More examples:

   - string passed to decrypt: `"asd dkjd"` -- string returned: `""`
   - string passed to decrypt: `"#evael#"` -- string returned: `"leave"`
   - string passed to decrypt: `"#t'nod##og#"` -- string returned: `"don't go"`
   - string passed to decrypt: `"tnod#od#ogce#og#foo#"` -- string returned: `"do go"`

- string passed to decrypt: `"#esaelp#sayt#esolc#are#eht#sd#rood#as"` --
  string returned: `"please close the door"`

Again, test this method by invoking it from main and printing the output. This time, you'll be able to make use of your test code!

3. Method **main()** (4 points).

   This method must prompt the user to enter a phrase, or to enter `"end"` to quit. Your check for the quit condition must be case insensitive, i.e., the user can enter `"enD"` or `"End"` or `"END"`, etc., to quit.

   If the user doesn't quit, invoke the **decrypt()** method and print the returned, decoded message, surrounded by parenthesis.

   Here's a sample interaction. User input, as usually, appears in boldface.

```
Please enter an encrypted message or the word 'end' to quit:  Hi there
Decoded message: ()

Please enter an encrypted message or the word 'end' to quit:
nalp#erugif#sdf#siht#so#tuo#
Decoded message: (figure this out)

Please enter an encrypted message or the word end to quit: End
Bye!
```

Grading: The grading schema for this project is roughly as follows:

- Your program should compile without syntax errors to receive any credit. If a part of your program is working, you will receive partial credit, but only if the program compiles without syntax errors.
- Correct implementation of each method is worth the number of points indicated above
- 2 points will be awarded for good programming style, as defined earlier.

Created by Tamara Babaian