CSULB
CECS 174
ProjectStrings

You are to write small multiple functions to manipulate strings, then call every function. Use output statements as needed to explain the function you are using and to show the output of the function or the returned value of the function.

Part I

Have the user input their full name and store it in a variable. Then, using string functions and string slicing operator, display the following information about the user's name:

1. Display the number of characters in the string.
2. Display the last character in the string.
3. Display the location of the first occurrence of the letter 'e' (0 if not found).
4. Display the number of words in the string.
5. Display the first word of the string.
6. Display the number of vowels in the string.
7. Display the string after capitalizing all vowels in the string. All consonants should be lower case.
8. Display the string centered between 50 '~'s, and 70 '+'s.
9. Display the string split in half on either end of 70 '*'s.

Use the string format function with placeholders to display your output.

Example (user's input is in italics):

Please enter your name: *John Jacob Jingleheimer Schmidt*

Your name is 31 characters long.

The last character is: t

The first 'e' is at position 17.

Your name has 4 words.

Your first name is John.

Your name contains 9 vowels.

Your name with uppercase vowels is: jOhn jAcOb jInglEhEImEr schmIdt

++++++++++~~~~~~~~~John Jacob Jingleheimer Schmidt~~~~~~~~~~++++++++++

John Jacob Jing***************************************leheimer Schmidt

Part II

Assign to a variable in your program a triple-quoted string that contains your favorite paragraph of text - perhaps a poem, a speech, instructions to bake a cake, some inspirational verses, etc.

Read to end of PartII before solving. You may need to do part L and M and use them as needed for Parts A to K.

In 1939 Ernest Vincent Wright published a 50,000 word novel called Gadsby that does not contain the letter “e.” Since “e” is the most common letter in English, that’s not easy to do.
http://www.holybooks.com/wp-content/uploads/Gadsby-by-Ernest-Vincent-Wright.pdf

In fact, it is difficult to construct a solitary thought without using that most common symbol. It is slow going at first, but with caution and hours of training you can gradually gain facility.

A- Write a function that **mirror**s its string argument, generating a string containing the original string and the string backwards.
B- Write a function that **remove**s all occurrences of a given letter from a string.
C- Write a function **char_num** that counts the number of alphabetic characters (a through z, or A through Z) in your text and then keeps track of how many are the letter ‘e’. Your function should print an analysis of the text like this: Your text contains 243 alphabetic characters, of which 109 (44.8%) are 'e'.
D- Modify the previous function to keep track of number of any character passed as a parameter are in the text. Try it with letter ‘e’. Did you get the same results?
E- Write a function called **no_e** that returns True if the given word doesn’t have the letter “e” in it.
F- Modify the previous function to return True if the passed word doesn’t have the character passed as another parameter. Call it for the same string and the letter ‘e’. Did you get the same results?
G- Modify your previous program to print only the words that have no “e” and compute the percentage of the words in the list have no “e.”
H- Write a function named **avoids** that takes a word and a string of forbidden letters, and that returns True if the word doesn’t use any of the forbidden letters.
  - Prompt the user to enter a string of forbidden letters and then print the number of words that don’t contain any of them. Can you find a combination of 5 forbidden letters that excludes the smallest number of words?

I- Write a function named **uses_only** that takes a word and a string of letters, and that returns True if the word contains only letters in the list. Can you make a sentence using only the letters acefhlo? Other than “Hoe alfalfa?”
J- Write a function named **uses_all** that takes a word and a string of required letters, and that returns True if the word uses all the required letters at least once. How many words are there that use all the vowels aeiou? How about aeiouy?
K- Write a function called **is_abecedarian** that returns True if the letters in a word appear in alphabetical order (double letters are ok). How many abecedarian words are there?

All of the functions in the previous section have something in common; they can be solved with a search pattern: Traverse a sequence and return when you find what you are looking for
The simplest example is:

L- Write a function **find** that takes a character and finds the index where that character appears. If the character is not found, the function returns -1.
M- Modify find so that it has a third parameter, the index in word where it should start looking.
N- Write a function called **is_sorted** that takes a **string** as a parameter and returns True if the string has the words **sorted in ascending** order and False otherwise.
O- Two words are anagrams if you can rearrange the letters from one to spell the other. Write a function called **is_anagram** that takes two strings and returns True if they are anagrams.

P- Write a function called **has_duplicates** that takes a string and returns True if there is any character that appears more than once in the string. **It should not modify the original list.**

Q- Write a function called **remove_duplicates** that takes a string and returns a new string with only the unique characters from the original.