Python Programming Assignment

Spring 2019 – Programming Languages

The purpose of this assignment is to test your Python skills in list comprehension and recursion. You will need to define a minimum the two functions listed below, although it is recommended to use helper functions. Those additional functions can be defined inside or outside of the two required functions.

Lipogram List Comprehension Function

Use Python's list comprehension to make a list of all the words in a string that have more than three letters and either begin with a vowel or contain any of the letters from a second string provided.

Name the function "*word_smith*". This function will receive two strings as input and must return the count of words in the list as an integer value. The first string will be used to build your list. The second string will be the one to check for characters.

Example:

Input strings: "Programming languages is definitely the best course I have taken in my entire undergraduate CS career!", "*Don't OVERTHINK it!*"
List: [Programming, languages, definitely, best, course, taken, entire, undergraduate]
Return: 8

Notes on vowel:

- Normal vowels including y.
- The case does not matter, A == a.

Notes on the input strings:

- Case matters, E =/= e.
- Spaces are not words so you can ignore them as characters.
- You should ignore any character that is not part of the English alphabet ([a-z,A-Z]) in the second string. Which means, from the example, that if the first input string contains the word "whomst?!", it would be included in the count because of the 'o' and 't', however, "play!" would not be included.
- Non-alphabet characters do not count towards the character count. Ee! is two characters.

Quaternary Conversion Recursion Function

Quaternary is the base-4 numeral system. It uses the digits 0, 1, 2 and 3 to represent any real number. In this function, you will convert a decimal number to its quaternary equivalent using recursion.

Decimal	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Quaternary	**0**	**1**	**2**	**3**	**10**	**11**	**12**	**13**	**20**	**21**	**22**	**23**	**30**	**31**	**32**	**33**

Name the function "*base_builder*". This function will receive an integer value as input and must return a tuple containing the sum of each digit in the quaternary value and the value itself. Both the sum and quaternary value in the tuple must be integer values.

Example:

Input value: 493
Quaternary value: 13231
Return: (10, 13231)

Notes:

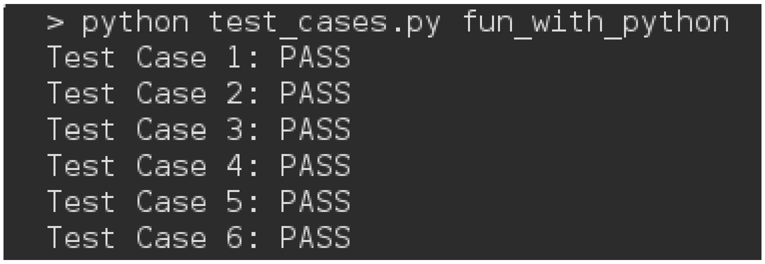
- You will not need to do floating point conversion.
- You will not need to convert negative numbers.
- Order of the tuple matters, (10, 13231) =/= (13231, 10).
- It is recommended to define your recursive function inside of the "*base_builder*" function.
- Check out https://www.mathsisfun.com/numbers/convert-base.php to confirm your decimal to quaternary conversions.

<u>Test cases</u>

Provide are a set of six test cases you can use to test your program. The test_cases.py file should be in the same directory as your solution python file. You can run the test case against your solution in the following manner:

1) Open terminal or command prompt and navigate to the location of the test case and your solution.
2) In the terminal, you will call the python application with the provided test_case.py file and your solution file without the .py extension as its parameters. In the example below, the name of my solution file is "*fun_with_python.py*".

> python test_cases.py <your solution file without .py extension>



```
> python test_cases.py fun_with_python
Test Case 1: PASS
Test Case 2: PASS
Test Case 3: PASS
Test Case 4: PASS
Test Case 5: PASS
Test Case 6: PASS
```

These will not be the test cases we will use to grade your assignment, and it is advised to create your own for testing.

<u>Rubric Information</u>

- You will be submitting a single .py file with your solution. You can name this file any way you like, but you should stick to the PEP8 naming conventions. https://www.python.org/dev/peps/pep-0008/
- Your functions are the only thing required for this assignment. You do not need a main function as we will be calling these functions directly.
- Your functions should only return the values of the output. Do not print anything to the screen.
- Your functions must use the required methods (list comprehension and recursion, respectively) to solve the problems. Not using these methods will result in a zero for that function.
- Your function names must be the same as listed for each specific problem. We will be using these names for our test cases. Incorrect function names will result in a loss of points.

Have fun and good luck!