Chapter9_Exercises

May 30, 2017

1 Chapter 9 Exercises

1.1 Exercise 9.1

Write a program that reads words.txt and prints only the words with more than 20 characters (not counting whitespace). (See the repo for words.txt)

```
In [3]: fin = open('words.txt')

        for line in fin:
            word = line.strip()

            if len(word) > 20:
                print (word)
```

```
counterdemonstrations
hyperaggressivenesses
microminiaturizations
```

1.2 Exercise 9.2

Write a function called has_no_e that returns True if the given word doesn't have the letter "e" in it. Modify your program from the previous section to print only the words that have no "e" and compute the percentage of the words in the list that have no "e".

```
In [30]: def has_no_e(s):
             for c in s:
                 if c == 'e':
                     return False
             return True

         fin2 = open('words.txt')
         total = 0
         count = 0

         for line in fin2:
```

```
    word = line.strip()

    total += 1

    if has_no_e(word):
        #print(word)
        count += 1
```

```
In [12]:    print('\n{:.2f}% of words do not have an e'.format(count*100/total))
```

```

33.07% of words do not have an e
```

1.3 Exercise 9.3

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. Modify your program to 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?

```
In [68]: def avoids(word,s):
             for c in word:
                 for c2 in s:
                     if c == c2:
                         return False
             return True

         usr = input('Enter a string of 5 forbidden letters: ')

         fin3 = open('words.txt')
         count2 = 0

         for line in fin3:
             if avoids(line.strip(), usr.lower()):
                 count2 += 1

         print(count2)
```

```
Enter a string of 5 forbidden letters: zxvqj
96067
```

1.4 Exercise 9.4

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?"

```
In [40]: def uses_only(word,s):
             testChars = "".join(set(s.lower()))
             tmp = 0
             for c in word.lower():
                 for c2 in testChars:
                     if c2 == c:
                         tmp += 1

             if tmp == len(word):
                 return True
             else:
                 return False

         print(uses_only('helloa', 'helloooo'))
         print(uses_only('USS Normandy', 'usnormandy '))
         print(uses_only('Try not. Do, or do not. There is no try.', 'trynot. d,hei
```

```
False
True
True
```

1.5 Exercise 9.5

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?

```
In [55]: def uses_all(word,s):
             testChars = "".join(set(s.lower()))
             wordNew = "".join(set(word.lower()))
             tmp = 0
             for c2 in testChars:
                 for c in wordNew:
                     if c2 == c:
                         tmp += 1
                 if tmp >= len(testChars):
                     return True

             return False

         print(uses_all('helloa', 'hellooooaaq'))
         print(uses_all('So say we all.', 'so ayweal'))
         print(uses_all('So say we all.', 'so aywealq'))
```

```
False
True
```

```
False
```

1.6 Exercise 9.6

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?

```
In [69]: def is_abecedarian(word):
             s = word.lower()
             for i in range(1, len(s)-1):
                 if not s[i+1] >= s[i]:
                     return False
             return True

         print(is_abecedarian('aa'))
         print(is_abecedarian('aacd'))
         print(is_abecedarian('aacb'))
         print(is_abecedarian('Now I am become Death, the destroyer of worlds.'))

         fin4 = open('words.txt')
         alphaTot = 0

         for line in fin4:
             if is_abecedarian(line.strip()):
                 alphaTot += 1

         print(alphaTot)
```

```
True
True
False
False
2493
```

1.7 Exercise 9.7

Give me a word with three consecutive double letters. I'll give you a couple of words that almost qualify, but don't. For example, the word committee, c-o-m-m-i-t-t-e-e. It would be great except for the 'i' that sneaks in there. Or Mississippi: M-i-s-s-i-s-s-ip-p-i. If you could take out those i's it would work. But there is a word that has three consecutive pairs of letters and to the best of my knowledge this may be the only word. Of course there are probably 500 more but I can only think of one. What is the word? (From Car Talk)

Write a program to find it.

```
In [119]: def has_three_consec(s):
              word = s.lower()
```

```
    tmp = 0
    index = 0
    first = True
    for i in range(0, len(s)-1):
        if word[i+1] == word[i]:
            if first:
                tmp += 1
                index = i+1
                first = False
            elif index == i - 1:
                tmp += 1
                index = i+1

    #print(tmp)

    if tmp >= 3:
        return True
    else:
        return False

def find_that_word():
    finja = open('words.txt')

    for line in finja:
        if has_three_consec(line.strip()):
            return line

    return None

print(has_three_consec('committee'))
print(has_three_consec('aaaaaa'))
print(find_that_word())
```

```
False
True
bookkeeper
```

1.8 Exercise 9.8

Here's another Car Talk Puzzler (http: // www. cartalk. com/ content/ puzzlers): "I was driving on the highway the other day and I happened to notice my odometer. Like most odometers, it shows six digits, in whole miles only. So, if my car had 300,000 miles, for example, I'd see 3-0-0-0-0-0. "Now, what I saw that day was very interesting. I noticed that the last 4 digits were palindromic; that is, they read the same forward as backward. For example, 5-4-4-5 is a palindrome, so my odometer could have read 3-1-5-4-4-5. "One mile later, the last 5 numbers were palindromic. For example, it could have read 3-6-5-4-5-6. One mile after that, the middle 4 out of 6 numbers were palindromic. And you ready for this? One mile later, all 6 were palindromic! “The question is, what was on the odometer when I first looked?” Write a Python program that tests all the six-digit numbers and prints any numbers that satisfy these requirements.

```
In [18]: def is_palindrome(word):
             return word[::-1] == word

         def find_that_number():
             i = 100000
             zeros = 0
             while i < 1000000:
                 test1 = i % 10000
                 zeros = 4 - len(str(test1))
                 s1 = '0'*zeros + str(test1)
                 test2 = (i+1) % 100000
                 zeros = 4 - len(str(test2))
                 s2 = '0'*zeros + str(test2)
                 test3 = (i+2)

                 if is_palindrome(s1) and is_palindrome(s2) and is_palindrome(str(t
                     print(i)
                     #print(s1)
                     #print(s2)
                     #print(test3)
                     #print('')
                 i += 1

         find_that_number()
```

```
999999
```

1.9 Exercise 9.9

Here’s another Car Talk Puzzler you can solve with a search (http: // www. cartalk. com/ content/ puzzlers): “Recently I had a visit with my mom and we realized that the two digits that make up my age when reversed resulted in her age. For example, if she’s 73, I’m 37. We wondered how often this has happened over the years but we got sidetracked with other topics and we never came up with an answer. “When I got home I figured out that the digits of our ages have been reversible six times so far. I also figured out that if we’re lucky it would happen again in a few years, and if we’re really lucky it would happen one more time after that. In other words, it would have happened 8 times over all. So the question is, how old am I now?” Write a Python program that searches for solutions to this Puzzler. Hint: you might find the string method zfill useful.

```
In [29]: def find_that_age():
             counter = 0
             for i in range(100):
                 if str(i).zfill(2) == str(i+18)[::-1]:
                     counter += 1
```

```
            if counter == 6:
                print(i)
    print(counter)

find_that_age()
```

```
57
8
```