paired_programming_activity_solutions

September 19, 2019

1 Paired programming activity

Question 1: Define a function char_counts that, given a string, counts and reports how many times each letter appears. Example string: "computing is too important to be left to men" (credit: Karen Sparck Jones). *Hint:* if you loop over a string, it goes through each character. *Challenge:* How short can you make your code and have it still work?

[1]:
```
from collections import Counter
def char_counts(s):
    return(Counter(''.join(filter(str.isalpha, s))))
```

[2]:
```
# Check your function
print('test: char_counts(\"computing is too important to be left to men\")')
string = "computing is too important to be left to men"
print(char_counts(string))
```

```
test: char_counts("computing is too important to be left to men")
Counter({'t': 7, 'o': 6, 'm': 3, 'i': 3, 'n': 3, 'e': 3, 'p': 2, 'c': 1, 'u': 1,
'g': 1, 's': 1, 'r': 1, 'a': 1, 'b': 1, 'l': 1, 'f': 1})
```

Question 2: Define a function maximum that takes two numbers as arguments and returns the largest of them. Hint: use if-then-else. (python has a max function built in, but write your own)

[4]:
```
def maximum(x,y):
    # takes 2 numbers as arguments and returns the largest of them
    if x > y:
        return(x)
    else:
        return(y)
```

[5]:
```
# Check your function
print('test: maximum(1,2)')
print(maximum(1,2))
print('test: maximum(2.1,2)')
print(maximum(2.1,2))
```

```
test: maximum(1,2)
2
```

```
test: maximum(2.1,2)
2.1
```

Question 3: Define a function max_of_three that takes three numbers as arguments and returns the largest of them.

[6]:
```
def max_of_three(x,y,z):
    # takes 3 numbers as arguments and returns the largest of them
    if x > y and x > z:
        return(x)
    elif y > z:
        return(y)
    else:
        return(z)
```

[7]:
```
# Check your function
print('test: max_of_three(1,2,3)')
print(max_of_three(1,2,3))
print('test: max_of_three(2.1,2,1)')
print(max_of_three(2.1,2,1))
print('test: max_of_three(1.1,2,1)')
print(max_of_three(1.1,2,1))
```

```
test: max_of_three(1,2,3)
3
test: max_of_three(2.1,2,1)
2.1
test: max_of_three(1.1,2,1)
2
```

Question 4: Define a function length that computes the length of a given list or string. (again, python has the len function, but write your own)

[8]:
```
def length(x):
    # takes a list or string and computes the length
    c = 0
    for i in x:
        c += 1
    return(c)
```

[9]:
```
# Check your function
print('test: length(\'mystring\')')
print(length('mystring'))
print('test: length([0,1,2,3,4,5])')
print(length([0,1,2,3,4,5]))
```

```
test: length('mystring')
8
test: length([0,1,2,3,4,5])
```

6

Question 5: Write a function is_vowel that takes a character (i.e. a string of length 1) and returns True if it is a vowel, False otherwise.

[10]:
```
def is_vowel(x):
    # takes a character and returns true if it is a vowel
    vowels = ['A','a','E','e','I','i','O','o','U','u']
    if x in vowels:
        return(True)
    else:
        return(False)
```

[11]:
```
# Check your function
print('test: is_vowel(\'A\')')
print(is_vowel('A'))
print('test: is_vowel(\'x\')')
print(is_vowel('x'))
```

```
test: is_vowel('A')
True
test: is_vowel('x')
False
```

Question 6: Define a sum_all function and a multiply_all function that sums and multiplies (respectively) all the numbers in a list of numbers. For example, sum([1, 2, 3, 4]) should return 10, and multiply([1, 2, 3, 4]) should return 24. (again, these exist in python but write your own)

[12]:
```
def sum_all(x):
    # takes a list of numbers and returns the sum
    s = 0
    for i in x:
        s += i
    return(s)

def multiply_all(x):
    # takes a list of numbers and returns the product
    p = 1
    for i in x:
        p *= i
    return(p)
```

[13]:
```
# Check your functions
print('test: sum_all([1,2,3,4])')
print(sum_all([1,2,3,4]))
print('test: multiply_all([1,2,3,4])')
print(multiply_all([1,2,3,4]))
```

```
test: sum_all([1,2,3,4])
```

```
10
test: multiply_all([1,2,3,4])
24
```

Question 7: Define a function reverse that computes the reversal of a string. So if you give it "this is too much" it will give you "hcum oot si siht"

[14]:
```
def reverse(x):
    # takes a string and returns the reverse of the string
    rev = ''
    for i in range(len(x)):
        rev += x[len(x)-i-1]
    return(rev)
```

[15]:
```
# Check your function
print('test: reverse(\'this is too much\')')
print(reverse('this is too much'))
```

```
test: reverse('this is too much')
hcum oot si siht
```

Question 8: Define a function is_palindrome that recognizes palindromes. You can test with "a man a plan a canal panama" or "radar".

[16]:
```
def is_palindrome(x):
    # takes a string and returns whether it is a palindrome
    x = x.replace(' ','')
    if x == reverse(x):
        return(True)
    else:
        return(False)
```

[17]:
```
# Check your function
print('test: is_palindrome(\'a man a plan a canal panama\')')
print(is_palindrome('a man a plan a canal panama'))
print('test: is_palindrome(\'radar\')')
print(is_palindrome('radar'))
```

```
test: is_palindrome('a man a plan a canal panama')
True
test: is_palindrome('radar')
True
```

Question 9: Write a function add_line_numbers that given a text file will create a new text file in which all the lines from the original file are numbered from 1 to n (where n is the number of lines in the file).

[18]:
```
import re
def add_line_numbers(file):
```

```
    # takes a file and makes a new file with lines numbered
    newfile = file.split('.',1)[0] + '_w_lines.txt'
    c = 0
    with open(file) as f:
        with open(newfile,'w') as out:
            for line in f:
                c += 1
                newline = str(c) + ' ' + line
                out.write(newline)
```

[19]:
```
# Check your function
print('test: add_line_numbers(\"paired_programming_testfile.txt\")')
add_line_numbers("paired_programming_testfile.txt")
```

```
test: add_line_numbers("paired_programming_testfile.txt")
```

Question 10: Write a function avg_word_len that given a text file will calculate the average word length of a text stored in a file (i.e the sum of all the lengths of the word tokens in the text, divided by the number of word tokens).

[20]:
```
def avg_word_len(file):
    # takes a file and returns the average length of the words in the file
    with open(file) as f:
        words = []
        for line in f:
            words.append(line.split())
    words = [x for sublist in words for x in sublist]
    word_lengths = [len(x) for x in words]
    print('Average word length:')
    print(sum(word_lengths)/len(words))
```

[21]:
```
# Check your function
print('test: avg_word_len(\"paired_programming_testfile.txt\")')
avg_word_len("paired_programming_testfile.txt")
```

```
test: avg_word_len("paired_programming_testfile.txt")
Average word length:
4.3515625
```