Advanced Functions Test

For this test, you should use the built-in functions to be able to write the requested functions in one line.

###Problem 1

Use map to create a function which finds the length of each word in the phrase (broken by spaces) and return the values in a list.

The function will have an input of a string, and output a list of integers.

In [1]:
```
def word_length(phrase):

    word_length = map(lambda x: len(x), phrase.split())

    return word_length
```

In [9]:
```
def word_lengths(phrase):

    y = map(lambda x: len(x), phrase.split())

    return y
```

In [2]:
```
phrase = ' i am yours'
phrase.split()
```

Out[2]:
```
['i', 'am', 'yours']
```

In [2]:
```
word_length('How long are the words in this phrase')
```

Out[2]:
```
[3, 4, 3, 3, 5, 2, 4, 6]
```

###Problem 2

Use reduce to take a list of digits and return the number that they correspond to. *Do not convert the integers to strings!*

In [3]:
```
def digits_to_num(digits):

    y = reduce(lambda x1,x2: x1*10+x2, digits)
    return y
```

In [5]:
```
digits_to_num([4,6,7,9,1])
```

Out[5]:
```
46791
```

In [4]:
```
digits_to_num([3,4,3,2,1])
```

Out[4]:
```
34321
```

###Problem 3

Use filter to return the words from a list of words which start with a target letter.

In [19]:
```
def filter_words(word_list, letter):

    def condition(word_list):
        for x in word_list:
            if x[0]==letter:
                return True

    y = filter(condition, word_list)
    return y
```

In [20]:
```
l = ['hello','are','cat','dog','ham','hi','go','to','heart']
filter_words(l,'h')
```

Out[20]:
```
['hello', 'ham', 'hi', 'heart']
```

In [13]:
```
l = ['hello','are','cat','dog','ham','hi','go','to','heart']
filter_words(l,'h')
```

Out[13]:
```
''
```

In [10]:
```
l = ['hello','are','cat','dog','ham','hi','go','to','heart']
filter_words(l,'h')
```

Out[10]:
```
['hello', 'ham', 'hi', 'heart']
```

###Problem 4

Use zip and list comprehension to return a list of the same length where each value is the two strings from L1 and L2 concatenated together with connector between them. Look at the example output below:

In [25]:
```
def concatenate(L1, L2, connector):

    x = connector*len(L1).split()

    #first_part = [x[n]+ L1[n] for n in len(L1)]

    #second_part = zip (first_part, L2)

    #return second_part

    return x
```

In [15]:
```
zip(['A','B'],['a','b'])
```

Out[15]:
```
[('A', 'a'), ('B', 'b')]
```

In [26]:
```
concatenate(['A','B'],['a','b'],'-')
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-26-8ba16c74b159> in <module>()
----> 1 concatenate(['A','B'],['a','b'],'-')

<ipython-input-25-0d31ce0c08e2> in concatenate(L1, L2, connector)
      1 def concatenate(L1, L2, connector):
      2
----> 3     x = connector*len(L1).split()
      4
      5     #first_part = [x[n]+ L1[n] for n in len(L1)]

AttributeError: 'int' object has no attribute 'split'
```

###Problem 5

Use enumerate and other skills to return a dictionary which has the values of the list as keys and the index as the value. You may assume that a value will only appear once in the given list.

In [31]:
```
def d_list(L):

    y = {}

    for item,value in enumerate(L):
        y[value]=item

    print y
```

In [32]:
```
d_list(['a','b','c'])
```

```
{'a': 0, 'c': 2, 'b': 1}
```

In [20]:
```
d_list(['a','b','c'])
```

Out[20]:
```
{'a': 0, 'b': 1, 'c': 2}
```

###Problem 6

Use enumerate and other skills from above to return the count of the number of items in the list whose value equals its index.

In [34]:
```
def count_match_index(L):

    x=0

    for item, value in enumerate(L):
        if item == value:
            x+=1
    return x
```

In [35]:
```
count_match_index([0,2,2,1,5,5,6,10])
```

Out[35]: 4

Great Job!