Built-in Functions Test Solutions

For this test, you should use built-in functions and 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_lengths(phrase):

    return list(map(len, phrase.split()))
```

In [2]:

```
word_lengths('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. For example, [1,2,3] corresponds to one-hundred-twenty-three.
Do not convert the integers to strings!

In [3]:

```
from functools import reduce

def digits_to_num(digits):

    return reduce(lambda x,y:x*10 + y,digits)
```

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 [5]:

```
def filter_words(word_list, letter):

    return list(filter(lambda word:word[0]==letter,word_list))
```

In [6]:

```
words = ['hello','are','cat','dog','ham','hi','go','to','heart']
filter_words(words,'h')
```

Out[6]:

```
['hello', 'ham', 'hi', 'heart']
```

Problem 4

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

In [7]:

```
def concatenate(L1, L2, connector):

    return [word1+connector+word2 for (word1,word2) in zip(L1,L2)]
```

In [8]:

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

Out[8]:

```
['A-a', 'B-b']
```

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 [9]:

```
def d_list(L):

    return {key:value for value,key in enumerate(L)}
```

In [10]:

```
d_list(['a','b','c'])
```

Out[10]:

```
{'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 [11]:

```
def count_match_index(L):

    return len([num for count,num in enumerate(L) if num==count])
```

In [12]:

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

Out[12]:

4

Great Job!