Week 5b Map, Filter, Reduce

Part 1 Map and Filter

In this exercise, try to come up with the answer without using IDLE/Python first. If there is an error, specify the cause and type of the error. Then type the expressions into IDLE to verify your answers. The objective is for you to understand why and how they work.

```
L = [9,2,1,3,4,5,6]
```

Expressions	**Output**
`map(lambda x: x > 2, L)`	
`list(filter(lambda x:x>2,L))`	
`tuple(map(lambda x: 'o' if x%2 else 'e',L))`	
`list(filter(lambda x: 'o' if x%2 else 'e',L))`	
`list(map(str,list(filter(lambda x:x%2,L))))`	
`str(list(filter(lambda x:x>30,map(lambda x:x*x,L))))`	

Part 2 Scale/Square Tuples

In the lecture, we talked about how to scale or square a list of numbers using ***our version*** of map(). How should we change the code for it to work on tuples?

```
>>> map(lambda x:x*x, L)
[81, 4, 1, 9, 16, 25, 36]

>>> map(lambda x:x*2, L)
[18, 4, 2, 6, 8, 10, 12]
```

Part 3 Sum Digits Square

We solved the "Sum Digits" problem of writing function `sum(n)` which returns the sum of all the digits in n before. Write an alternative implementation `sds(n)` using ***our version*** of map() and built-in function sum().

How about "Sum Digit Square"? Write a function `sdss(n)` which returns the sum of the square of every digit in n.

Part 4 Taylor Series

The Taylor series is an infinite sum of many terms, e.g.

$$\sin x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots \quad \text{for all } x$$

$$\cos x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \cdots \quad \text{for all } x$$

$$\tan x = \sum_{n=1}^{\infty} \frac{B_{2n}(-4)^n(1-4^n)}{(2n)!} x^{2n-1} = x + \frac{x^3}{3} + \frac{2x^5}{15} + \cdots \quad \text{for } |x| < \frac{\pi}{2}$$

How do we use **our version** of map() and sum() to compute the above without using loops or recursion?

Part 5 Reduce

```
def reduce(f, seq):
    if not seq:
        return seq
    first = seq[0]
    for i in seq[1:]:
        first = f(first,i)
    return first
```

Predict the output of the following function call:

```
reduce(lambda x,y:x+y,[1,2,3,4])
```