**Python: map, reduce and filter**

***Map function***

Example of using map to convert a ***list*** of temperatures from Celsius to fahrenheit. It is an iterator.

def cels\_to\_fhr(c):

return (9/5)\*c + 32

temps = [21, 65, 38, 42]

list(map(cels\_to\_fhr, temps))

returns: [69.80000000000001, 149.0, 100.4, 107.60000000000001]

Consider having a ***list of tuples*** with city names and temperature in (C) which must be converted to Fahrenheit. We can use the lambda expression:

temps = [('berlin',21), ('sudan',65), ('dubai',38)]

c\_to\_f = lambda data:(data[0], (9/5)\*data[1] + 32)

list(map(c\_to\_f, temps))

returns:

[('berlin', 69.80000000000001), ('sudan', 149.0), ('dubai', 100.4)]

***Filter function***

Consider choosing items above the average in a list:

import statistics

data = [1.3, 2.7, 0.8, 5.6, 2.1, 8.5]

avg = statistics.mean(data)

print(avg)

returns: 3.5

list(filter(lambda x: x > avg, data))

returns: [5.6, 8.5]

Another example of using the filter function:

countries = ['', 'France', 'Argentina', '', 'Belgium', '', 'Brazil']

list(filter(None, countries))

returns: ['France', 'Argentina', 'Belgium', 'Brazil']

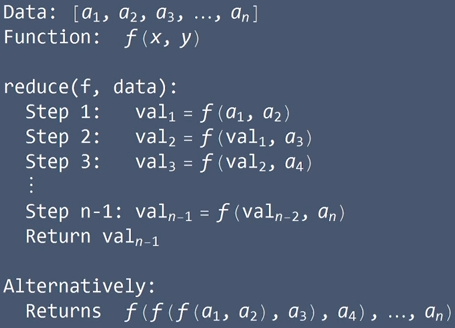
***Note:*** in python false values include: “”, 0, 0j, [], {}, (), None

(empty strings, empty tuples, empty lists, empty dictionaries, …)

***Reduce function***

It has been moved to the functools module in Python 3

For a given data and function (f), it works the following way:



from functools import reduce

numbers = [5, 4, 9, 6, 2, 3]

multiply = lambda x, y:x\*y

reduce(multiply, numbers)

returns: 6480