Exercise 4 – Functional Programming for Data Analysis

Objective

To write and call our own user-written functions, potentially as part of a larger data analysis pipeline.

Questions

1. Create a function which takes two arguments: a function and a list. The list should contain a list of names in the format:

names = ["Ellery Queen", "Sherlock Homes"]

The function should be called “flatMap” and it should be possible to call it like so:

flatMap(str.split, names)

Your function should process this list of names and return a list in the following format (based on this test data):

['Ellery', 'Queen', 'Sherlock', 'Homes']

**If time allows:**

* Try to include suitable type hints and a docstring for your function.
* Consider writing this solution using a different technique…

1. Using python's in-built transformation functions, start with a list of strings, and produce:
2. a list of bools whether there is data or not
3. a list of string lengths
4. a list of string lengths more than 3 characters

Here is a sample list of strings:

["ABC", "abc", "", "1", "!?%$"]

Note. There are many ways to achieve these solutions – try to experiment with different techniques.

Solutions

**Question 1**

Our function:

from typing import List, Callable

def flatMap(f:Callable, data:List)->List:

"""

Queue[String] -> Queue[String] via f: String -> Queue[String]

C[A] -> C[B] via f: A -> C[B]

"""

new = []

for e in data:

many = f(e)

for m in many:

new.append(m)

return new

if \_\_name\_\_ == "\_\_main\_\_":

names = ["Ellery Queen", "Sherlock Homes"]

result = flatMap(str.split, names)

print(result)

**Question 2**

**a.**

def is\_data(value:str)->bool:

"""

Checks to see if string is not empty

"""

return len(value)>0

my\_list = ["ABC", "abc", "", "1", "!?%$"]

print(list(map(is\_data, my\_list)))

**produces: [True, True, False, True, True]**

**b.**

my\_list = ["ABC", "abc", "", "1", "!?%$"]

print(list(map(len, my\_list)))

**produces: [3, 3, 0, 1, 4]**

**c.**

def has\_length(value:str)->str:

return len(value) > 3

my\_list = ["ABC", "abc", "", "1", "!?%$"]

print(list(filter(has\_length, my\_list)))

**produces: ['!?%$']**