Lab #3

Due: 10/24/22 @ 11:59pm

Content Covered
Ordered and Associative Collections in Python and JavaScript

Getting Started

1. Sign into replit.com
2. Create a new Python REPL for problems 1-3, and a new JavaScript (node.js) REPL for problems 4-6

Editing Your Code

In this lab you will be defining functions to solve 6 problems as described below. The first three problems will be implemented in Python in `main.py`, and the last three will be implemented in JavaScript in `index.js`. Feel free to solve the problems in any order of your choosing. It is important that you name your functions exactly as specified by each problem so that the autograder will be able to call them.

Any variables and parameters can be named however you like. You can write any additional functions to help you solve the problems as you see fit. Each problem is more complex than the ones before it.

Submission

The Autolab submission for this lab will open up no later than 10/17/22 @ 5:00pm.

Your final submission for this lab will be two .zip files downloaded from replit.com and submitted to Autolab. The first three problems will be submitted to ***Lab #3 - PY***, and the last three will be submitted to ***Lab #3 - JS***. You may submit as many times as you like, but only your last submission will count for your final grade.

Submissions close at 11:59pm on 10/24/22 and no late submissions will be accepted.

Problems

Problem 1

Define a Python function named `filterNegative` which takes a list of numbers as input and returns a new list containing only the non-negative values, preserving their relative order. When called with an empty array as the argument the function must return a new empty array.

For example, `filterNegative([3,-6,5,-2])` must return `[3,5]`.

Problem 2

Define a Python function named `sumMap` which takes two dictionaries and returns a single combined dictionary. For keys which exist in only one dictionary, they should appear in the returned dictionary unchanged. For keys which exist in both dictionaries, the value in the returned dictionary should be the sum of the values in the input dictionaries.

For example, `sumMap({"ball":4, "apple":6}, {"orange":9, "apple":10})` must return `{"ball":4, "apple":16, "orange":9}`

Problem 3

Define a Python function named `groupGrades` which takes a list of numbers as input and returns a dictionary where the keys are letter grades based on the table below, and the values are lists of the grades from the original list that correspond to that letter grade.

Numeric grade	**Letter grade**
90 <= grade <= 100	A
80 <= grade < 90	B
70 <= grade < 80	C
60 <= grade < 70	D
0 <= grade < 60	F

For example `groupGrades(100,97,72])` must return `{"A":[100,97], "C":[72]}`.
Hint: You may find a function you defined as part of Lab #1 useful for this problem…

Problem 4

Solve Problem 1 in JavaScript.

Problem 5

Solve Problem 2 in JavaScript.

Problem 6

Solve Problem 3 in JavaScript.

Useful Suggestions and Advice

- **Think before you code** - think about the input to the function, and the desired output, and plan out how you would solve the problem in your head or on paper before you start to code. Having an idea of where you are going before you start typing can make the coding process smoother.
- **Write your own tests** - each problem describes an example test case or two that you can use to check if your implementation is correct. But they are not sufficient to guarantee you have a completely correct implementation. It can help to think through and write additional test cases (even before you code) to make sure your functions are behaving as you expect them to. This will also allow you to more quickly test your code as you go, rather than having to wait for Autolab to be open, and to go through the entire submission process just to get feedback.
- **Break down bigger problems into smaller ones** - if you are stuck on a problem, try solving a smaller piece of the problem first. You can even write it as a separate function, and test it out. Then once you have the small pieces working, building the solution to the bigger problem from your smaller solutions should be easier.
- **Ask for help** - during lab sessions, in office hours, and on Piazza. If you do choose to ask questions on Piazza **do not post your own code in public posts.** If you are going to make your question public, make sure it is general/conceptual in nature. Otherwise, make your question private to just the instructors.