

COMP 690 Data Fusion Fall 2009 Programming Assignment 1

Due in the digital drop box by 11:00 PM on Wednesday, Sept. 9

1. Write a Python script in which you implement the following list functions. The general idea is to work on a list of integers that you read in.

- Input a list of integers (using `raw_input()`).
- Output a list (of integers) using a good format.
- Passed an integer, return the integer in the list that is closest to it. Break draws any way you please.
- Sort the list.
- Add a given integer to each list element.
- Given a second list, extend the first with it.
- Given an integer, remove all elements less than that integer.

Some of these functions simply wrap calls to builtin Python functions.

The script should contain a loop that repeatedly outputs a menu that allows the user to choose one of these functions other than the input and output functions, to choose a specified operation using the input or output function, or to exit the loop. A variable will hold the value of the list being worked on. The two operations using the input or output function that can be chosen from the menu are as follows.

- Input a list and assign it to this variable. (This is how an initial value is obtained, but it also can be used to start with a completely new list, overwriting the old.)
- Output the list that is the value of this variable.

Sometimes the user will have to be prompted for values to be passed to the selected function. The functions themselves, other than the input and output functions, should not read values from or write values to the terminal. Some of the functions modify the list, while others produce a new list, used to overwrite the value. In either case, we'd expect the user to output the new list after performing the operation by selecting the output operation from the menu.

2. Write a Python script that reads in names and corresponding bank balances (integers, without a '\$' or commas) and makes a dictionary of the associations. You can prompt the user first for how many name-income pairs (s)he wants to enter. Use `raw_input()` for input. After constructing the initial balances, repeatedly do the following:

Prompt for a transaction for each bank customer: a deposit, a withdrawal, or no transaction. Update the dictionary of bank balances and output the updated customer-balance associations. Make a dictionary of the transactions where the keys are the same as for the balance dictionary; positive values represent deposits, negative values represent withdrawals, and a zero represents no transaction.

Maintain a list of the transaction dictionaries (one for each iteration). When the loop exits, output the current balances and the list of transactions in a well-formatted table. This could be implemented with recursion as well as with a loop.