Austin McCalley

Dr. Jennifer Parham-Mocello

Computer Science 160-020

14 November 2019

Assignment 8

Orientation

The terms which I am going to be investigating is object-oriented programming, cache, compiler, and interpreter. The term object-oriented programming or OOP is a way of programming which deals with working with objects which is a type of data structure. For example, when you define a list it creates an object which contains all the functions which can be used to operate on that list; in python, the result of the len() function is contained in this object. The term cache is to refer to supplemental memory which can be used to access repeated information faster. For example, when calculating the Fibonacci sequence at some index there are base numbers which are calculated a lot and it could speed up the equation if you add that number to a cache to have it saved in memory for future access to speed up calculations. A compiler is a program which processes the syntax of the code and converts it into a machine language for the computer to use. An example of this is the compiler software g++ which compiles C++ into machine code for the computer to understand. An interpreter is very similar to a compiler but instead of generating machine code it directly executes the code which is written. For example, when writing in Python the interpreter directly translates it into a machine language and has the computer execute it.

Understanding the Problem

The problem which we are trying to solve is asking for three different inputs. The first input is a list of names, meanwhile, the second and the third inputs are going to be a list of positive integers. The program is going to calculate the number of occurrences each letter occurs in all the names. For example, if the input of names is Frank and Brian, the frequency will have 1 f, 2 r's, 2 a's, 2 n's, 1 k, 1 b, and 1 i. The program is then going to run five different functions on the two lists of integers. The first function will be returning a true or false boolean if the lengths of the two lists. The second function will add up all the integers in the list and return the sum. The third function will find the average between all the numbers in the list. The fourth function will return a true or false boolean if the lengths of the lists are different. And finally, the fifth function will return the common numbers between the two inputs. The function will do all the error handling prior to performing any operations and there will be no variables defined outside of a functions scope, in the global scope. The program will output the results of all five of the functions that are performed on the two lists of integers.

Devise a Plan

In this program I am going to make two different functions to take the inputs of the names and the integers. The names input function will continue to ask for a name until exit is typed and then return all the names that were returned. The numbers input will ask for the list of numbers to be a comma separated all on one line, but the function will be called twice. The numbers input function will remove all non-integer from the function and int cast them post error checking. As well, when I am computing the five functions for the two lists of numbers, I am

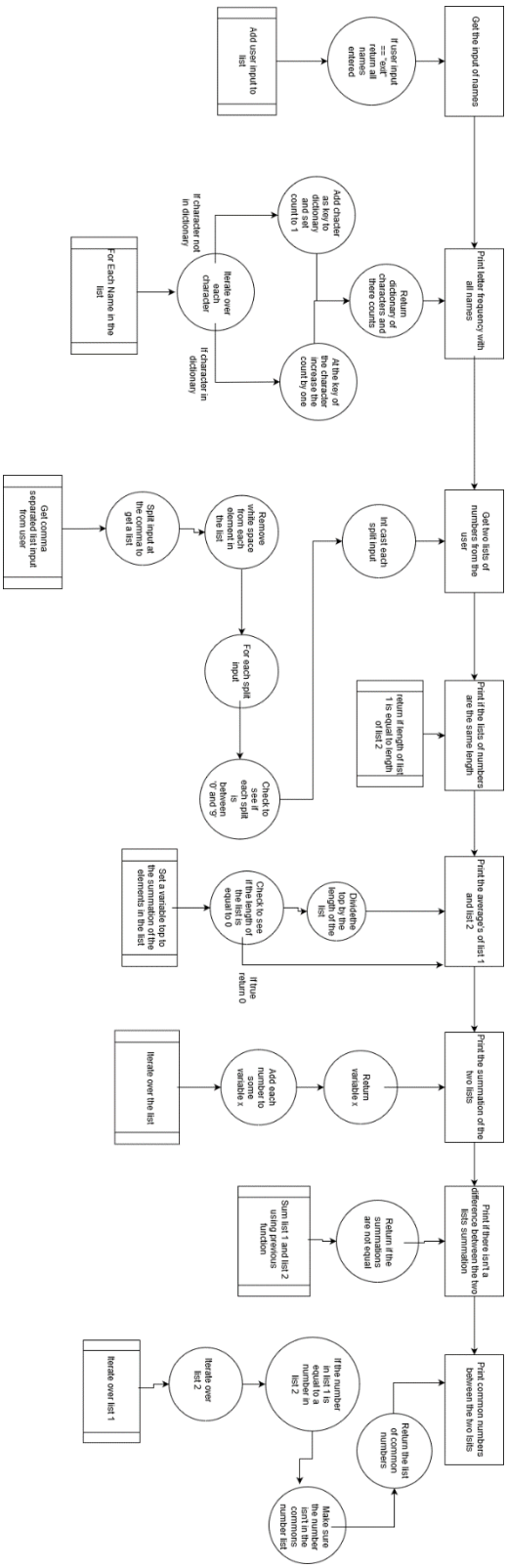
going to have them called in one main function to streamline the process. As well I am going to describe what is being printed rather than just randomly printing data to the screen.

I am going to use a dictionary to find all the occurrences of letter in the list of names. I am going to iterate through each name in the list of names and then iterate over each character. I am going to make the character lowercase and then see if it exists yet in the dictionary, if it does exist in the dictionary already then I am going to increase the count by one on that key(character) if it does not exist in the dictionary yet I am going to the key equal to the character and then set the initial value to be one. This will return a dictionary of the count of each character in all the names there were inputted from the user.

To find common numbers between the two lists of integers I am going to have an array which is empty which will hold all the common integers. Then I will iterate over each value in one of the lists and then for each value of that it will iterate over the entire next list and see if any of the numbers are equal and not already in the list of common integers. This is a slow process for big lists of numbers but still will work. This function will return a list.

I am going to handle bad input by using the integer checker that we built in the 6^{th} lab. As well when I am parsing the string input of the two different lists of numbers, I am going to split the string at the comma which will return an array based off the string. Then I am going to iterate over that array and use the trim() function to remove all whitespace from the numbers and then use the integer checker to ensure that it is an int. Then I will return the new list of just int casted integers which made it through all the checks.

Flow Chart


Get the input of names
If user input == "exit" return all the names entered
Add user input to list
Print letter frequency with all names
Return dictionary of characters and there counts
Iterate over each character
If character not in dictionary
Add charater as key to dictionary and set count to 1
If character in dictionary
At the key of the character increase the count by one
For Each Name in the list
Print if the lists of numbers are the same length
return if length of list 1 is equal to length of list 2
Get two lists of numbers from the user
Int cast each split input
Remove white spaces from each element in the list
Split input at the comma to get a list
Get comma separated input from user
For each split input
Check to see if each split is between 0 and 3
Print the average's of list 1 and list 2
Divide the top by the length of the list
Check to see if the length of the list is equal to 0
If true return 0
Set a variable top to the summation of the elements in the list
Print the summation of the two lists
Return variable x
Add each number to some variable x
Iterate over the list
Print if there isn't a differences between the two lists summation
Return if the summations are not equal
Sum list 1 and list 2 using previous function
Print common numbers between the two lists
Return the list of common numbers
Make sure the number isn't in the common number list
If the number in list 1 is equal to a number in list 2
Iterate over list 2
Iterate over list 1

Looking Back

When looking back upon my code for this assignment I realize that list and string manipulation is very important when you are working with different problems. The hardest part for me during this project was the frequency of letters and common numbers because I had to think in the aspect of comparing different things between two lists.

Frequency of Letters in Names

Input	Expected	Actual	Pass
Frank, Bob, George, Jill, Larry	2 a, 2 b, 2 e, 1 f, 2 g, 1 I, 1 j, 1 k, 3 l, 1 n, 2 o, 4 r, 1 y	2 a, 2 b, 2 e, 1 f, 2 g, 1 I, 1 j, 1 k, 3 l, 1 n, 2 o, 4 r, 1 y	**Yes**

Summation

Input	Expected	Actual	Pass
1,2,3,9,8,7	30	30	**Yes**
1,2, -3	0	0	**Yes**

Summation Comparison

Input #1	Input #2	Expected	Actual	Pass
1,2,3	4,2	The sums are the same so False that they are different	False	**Yes**

Lists are Same length

Input #1	Input #2	Expected	Actual	Pass
1,2,3	99,345,32	True	True	**Yes**

Average of the list

Input	Expected	Actual	Pass
1,2,3	2	2.0	**Yes**
Null input	0	0	**Yes**

Common Numbers between Lists

Input #1	Input #2	Expected	Actual	Pass
1,2,3	3,4,5	3	3	**Yes**