**Lab 15 - More on functions**

**Questions:**

**1.** What is the difference between an argument and a parameter?

**2.** What does the following code print?

**def** my\_function(b\_list):

b\_list[0] = 100

a\_list = [1,2,3]

a\_list = [5,6,7]

my\_function(a\_list)

print(a\_list)

[100, 6, 7]

**3.** What does the following code print? Explain.

def f(a, b=2):

pass

f(a=3, b=4)

print(a, b)

An error because a was never assigned a value

**4.** Consider the following code. The “print(x)” statement generates an error. Provide two different ways to print “x” properly.

def add\_one(number):

x=1

number = number + x

print(number)

add\_one(3)

print(x)

1. Print(x) inside the function(after it has been declared)
2. Return x at end of function and print the call of the function (print(add\_one(3)))

**5.** Give the output indicated for the following program. It will be helpful to draw diagrams to indicate the relationships among arguments, parameters, and their objects. Think about which arguments are mutable and which are immutable.

**def** func1(list1, list2, str1):

**if** len(list1) > 3:

list1 = list1[:3]

list2[0] = **'goodbye'**

str1 = **''**.join(list2)

arg1\_list = [**'a'**, **'b'**, **'c'**, **'d'**]

arg2\_list = [**'hello'**, **'mother'**, **'and'**, **'father'**]

arg\_str = **'sister'**

func1(arg1\_list, arg2\_list, arg\_str)

print(arg1\_list) *# Line 1*

print(arg2\_list) *# Line 2*

print(arg\_str) *# Line 3*

(a) What output is produced by Line 1 when the program is executed?

Line 1: ['a', 'b', 'c', 'd']

(b) What output is produced by Line 2 when the program is executed?

Line 2: ['goodbye', 'mother', 'and', 'father']

(c) What output is produced by Line 3 when the program is executed?

Line 3: sister

**6.** Remove odds or evens:

(a) Write a function that takes a list of integers as an argument, removes even numbers from the list, and returns the modified list.

(b) Write a function that takes a list of integers as an argument, removes odd numbers from the list, and returns the modified list.

(c) Write a function that takes a list of integers and a Boolean as arguments. If the

Boolean is True, the function removes odd numbers from the list; otherwise, evens

are removed. The function returns the modified list.

**7.** Data mining is the process of sorting through large amounts of data and picking out relevant information. It is usually used by business intelligence organizations and financial analysts but is increasingly being used in the sciences to extract information from the enormous data sets generated by modern experimental and observational methods. In this project, we want to do some preliminary data mining to the prices of some company’s stock. So that we can speak in specifics, let’s look at Google. Your program will calculate the monthly average prices of Google stock from November 2018 to November 2019 and tell us the six best and six worst months for Google.

We provide the data reading function; you write the next two and a main that calls the functions.

Hints:

1. The list sort() and reverse() methods will be useful. Experiment with how they sort a list of tuples—notice how they sort on the first item.

2. To create a tuple, put items in a comma-separated list with parentheses:(x,y).

3. When working with a list of lists (or a list of tuples), the first item in the first list is some list[0][0] and the second item in that same first list is someList [0][1]

.

(a) First you need a history of stock prices. The file GOOG.csv in brightspace can be used for that. The file format is indicated by the first few lines:

Date,Open,High,Low,Close,Volume,Adj Close

2008-09-19,461.00,462.07,443.28,449.15,10006000,449.15

2008-09-18,422.64,439.18,410.50,439.08,8589400,439.08

(b) get\_data\_list(file\_name: string) -> list

The csv file is a comma-separated file, so we can split the data on commas. The following function will read a file, split the lines in the file on commas, and put the data into a list that is returned. The result is a list of lists where each line is a list. Also, every item is a string. To read our file, call it using our file name: get data list('table.csv'). Experiment with this function in the shell to get a sense of what is returned.

**def** get\_data\_list(file\_name: string) -> list:

*"""Receives a CSV file name in string format and returns a*

*list of lists. Each list corresponds to a row in the file.*

*"""*

data\_file = open(file\_name, **"r"**)

data\_list = [] *# start with an empty list*

*# strip end-of-line, split on commas, and append items to list*

**for** line\_str **in** data\_file:

data\_list.append(line\_str.strip().split(**','**))

**return** data\_list

c) get\_monthly\_averages(data list)

In this function, you will use the data list generated by the get data list function as the

parameter. Use the Date, Volume, and Adj Close fields to calculate the average monthly prices. Here is a formula for the average price for a month, where Vi is the volume and Ci is the day’s adjusted close price (Adj Close).

average price = (V1 ∗ C1 + V2 ∗ C2 + ... + Vn ∗ Cn)/(V1 + V2 + ... + Vn)

For each month create a tuple with two items: the average for that month and the date (you need only the month and year). Append the tuple for each month to a list (e.g., monthly averages list), and after calculating all the monthly averages, return this list. We use tuples here because once these values are calculated we don’t want to accidentally change them!

(d) print\_info(monthly\_averages\_list)

In this function, you need to use the list of monthly averages calculated in the get monthly averages function. Here you will need to find and print the six best (highest average price) and six worst (lowest average price) months for Google’s stock. Print them in order from highest to lowest and print to two decimal places. Format the output so that it looks good and include informative headers. This function does not return anything.

(e) If you don’t call these functions, they are useless. Thus, you should write code to call them. Write a main function def main(): that is calls the other function in the correct other. Then make a single call to main to run the code.