Module 4 Assignment

Simplifying Your Code: Functions and How to Use Them

# Instructions

Create a jupyter notebook called ‘module4\_assignment’ and use it to answer the homework problems below.

Remember the following while working on your assignment:

* Make sure your functions return the request result. The functions should use the return keyword to return the result. The function should not print the result.
* Do not import any other libraries to solve the problems. I want you to use the basics you have learned to solve the following problems. All of these can be solved with the basics. As always, there are multiple ways to solve these problems, but the goal here is to master the basics.
* Use good variable names. For example, if you are using a variable to store a list of words, a good variable name would be ‘word\_list’. Some bad variable names would be ’word’, ‘string’, ‘list’ or ‘X’.
* Remove any testing code before you turn in the assignment. This means removing any code that is outside of your functions.

# Homework Problems

## Problem 1

Define a function called division. This function accepts two arguments, which are numbers, and divides the first argument by the second. This is a simple function, but it will help us practice creating functions. Here are some examples of how the function should work:

### Example 1

|  |
| --- |
| result = division(3, 5)  print(result) # 0.6 is the expected result |

### 

### Example 2

|  |
| --- |
| result = division(1, 7)  print(result) # 0.14285... is the expected result |

## 

## Problem 2

Define a function called **multiply\_numbers**. The function accepts one argument which will be a tuple of numbers. The function will multiply all of the numbers together and return the result.

### Hints

* Remember you can iterate through the numbers in the tuple using a for loop.
* Make sure you pass a tuple of numbers, not a list, or one number. An example of a tuple of numbers is (1, 2, 3, 4, 5)
* The start of your functions definition should look like the following line (you do not have to use ‘my\_tuple’ for the argument name)

|  |
| --- |
| **def** **multiply\_numbers**(my\_tuple): |

You can test your function with the following code:

|  |
| --- |
| my\_tuple = (4, 5, 6)  result = multiply\_numbers(my\_tuple)  print(result) # '120' is the expected result |

## Problem 3

Define a function called **filter\_list.** This function will take two arguments. The first argument is a list of strings. The second argument is a string that the function will filter out of the list. The function should return a list that contains all of the items in the list input to it, except for any items that were equal to the second argument. Here are some examples of inputs and expected outputs for the function:

### Example 1

|  |
| --- |
| filter\_string = 'hello'  string\_list = ['cat', 'hello', 'dog', 'hello']  result = filter\_list(string\_list, filter\_string) print(result) # '['cat', 'dog']' is the expected result |

### Example 2

|  |
| --- |
| filter\_string = 'apple'  string\_list = ['apple', 'apple', 'pear', 'apple']  result = filter\_list(string\_list, filter\_string) print(result) # '['pear']' is the expected result |

### Hints

* You can use a for loop to loop through the list
* You can create a new list, that is the list that the function returns

## 

## Problem 4

Define a function called **longest\_word.** The function takes one argument that is a list of words. The function should return the longest word in the list. Here are some examples of how the function should work:

### Example 1

|  |
| --- |
| word\_list = ['apple', 'orange', 'banana']  result = longest\_word(word\_list)  print(result) *# 'orange' is the expected result* |

### Example 2

|  |
| --- |
| word\_list = ['cat', 'lama', 'penguin']  result = longest\_word(word\_list)  print(result) *# 'penguin' is the expected result* |

## Problem 5

Define a function called **list\_to\_unique.** This function accepts one argument, which is a list of anything (number, string, etc…). The function should return a list of only the unique items (no duplicates). The list that is returned should have the items in the same order as they were in the input list. Here are some examples of how the function should work:

#### Example 1

|  |
| --- |
| input\_list = [5,5,'apple','pizza','apple']  result = list\_to\_unique(Input\_list)  print(result) # '[5,'apple','pizza']' is the expected result |

#### Example 2

|  |
| --- |
| input\_list = ['florida','ohio','ohio','california','california','florida']  result = list\_to\_unique(input\_list)  print(result) # '['florida','ohio','california']' is the expected result |

## How Turn In The Assignment

First, remove any testing / scripting code so that only the functions definitions remain in your notebook. If you want to make a copy of the notebook to keep a copy of any scripting or testing code that is fine. But, otherwise - ONLY turn in the functions definitions.

Then, please download the notebooks as a Python file (Go to the file menu, in Jupyter notebooks, and choose "Download as…", then choose python to download as a python file) and submit the assignment.