"Introduction to Programming" - Lab 5

Exercise Sheet – Practice Session (our last practice session)

This exercise sheet examines the concepts covered in recent lectures (as well as content we may cover this week).

Setup:

1. Note: anywhere I write <text> (text enclosed in a pair of angle brackets), I want you to change this text to the text I'm looking for (making sure to remove the angle brackets), so for example when I say:

    a. # <todays date>, you write:
    b. # Tuesday 12-10-2021
2. Create a new Folder called "Programming_Folder", if it does not exist. This will be your course Project folder.
3. Download from Canvas the zip file " Lab<number>.zip", unzip and place the contents in the "Programming_Folder" folder.
4. Rename Lab<number> to the number of this Lab
5. The unzipped folder contains, two Python files:
    a. my_functions.py
        i. my_functions.py - is where you will place all the functions created for this lab
    b. main.py
        i. main.py - is the primary file you will call when testing your code
        ii. all of the functions created in 'my_functions.py', will be called/invoked from your 'main.py' file
    c. In these files, I give one example where I create a function called 'print_function()' in my_functions.py and I call it from main.py
6. You should understand all of the code in these two files, if not ask me and I will explain
7. Make sure you add your name and student number to both 'main.py' and my_functions.py'
8. Once complete, and prior to the submission deadline, please upload your updated version of 'functions.py' to Canvas. Do not upload your version of 'main.py', as I will test the code in your 'my_functions.py' file, with my own code.

Exercises:

1. Pass a number, as parameter *num1*, to a function called *fizz_buzz*() For multiples of 3, return “Fizz”. For multiples of 5, return “Buzz”. For numbers which are multiples of both 3 and 5, e.g., 15, 30, etc, return “FizzBuzz”. If none of the conditions are true, simply return the number itself. Return the error statement 'Input value(s) must be a number’ when the inputs are not integers. Examples of what is returned by the function call are shown:

    a. fizz_buzz(3) -> 'Fizz'
    b. fizz_buzz(5) -> 'Buzz'
    c. fizz_buzz(15) -> 'FizzBuzz'
    d. fizz_buzz(14) -> 14
    e. fizz_buzz('a') -> 'Input value(s) must be a number'

2. I want you to rewrite the *fizz_buzz()* function from item 1. In the original function, a number was passed to the function. For multiples of 3 the function returned “Fizz”. For multiples of 5, returned “Buzz”. For numbers which are multiples of both 3 and 5 returned “FizzBuzz”. I want this functionality to remain in the new function (remember default values).

    I now want you to add two additional parameters, namely *divisor_1* (parameter 2) and *divisor_2* (parameter 3). Now the function requirements are: For multiples of *divisor_1* the function return “Fizz”. For multiples of *divisor_2*, return “Buzz”. For numbers which are multiples of both *divisor_1* and *divisor_2* return “FizzBuzz”. Examples of what is returned by the function call are shown:

    a. fizz_buzz(3) -> 'Fizz'
    b. fizz_buzz(5) -> 'Buzz'
    c. fizz_buzz(15) -> 'FizzBuzz'
    d. fizz_buzz(14) -> 14
    e. fizz_buzz('a') -> 'Input value(s) must be a number'
    f. fizz_buzz(4, 4, 6) -> 'Fizz'
    g. fizz_buzz(6, 4, 6) -> 'Buzz'
    h. fizz_buzz(15, 3, 5) -> 'FizzBuzz'

3. I want you to rewrite the *grades()* function from Lab 4, adding a parameter called *number*. In the original function, you passed a number to the function, which would return the corresponding *Letter Grade*. The function will now also take a *Letter Grade* and should return the *Numerical Grade* range. Examples of what is returned by the function call are shown:

    grades(86) -> "A" (the string A is returned)

    grades("A") -> "85-100" (the string 85-100 is returned)

    grades(110) -> "The input numerical grade is outside the range supported"

grades("H") -> "The input letter grade is outside the range supported"

Update error message handling in the new function to handle the different inputs (if not str or int) – *'Input value must be a number or a letter'*.

4. For *fizz_buzz*(), I want you to add error handling (exception handling) to catch those casting issues, or non-type issue, and use this to return the error message, *'Input value(s) must be a number'*, for the function.

Error Checking:

1. There is no error checking in this lab, but...
    a. I want you to try and break your code :)
    b. Change the inputs to the wrong type, the wrong value, and document what happens as a comment in your code.
    c. Try the break challenge for each function you have created. For now you only need to try to break each function once, and then fix for this once.