EMAT10007 – Introduction to Computer Programming

Exercises – Week 4. Functions

Reminder: The exercise sheets use `<?>` as a placeholder to represent something missing - this could be an operator, a variable name, function name, etc. Multiple `<?>` in the same question are not necessarily representing the same thing.

Part 1. Functions in Python

Exercise 1 - 99 Bottles of beer

1. 99 Bottles of Beer" is a traditional song in the United States and Canada. It is popular to sing on long trips, as it has a very repetitive format which is easy to memorise, and can take a long time to sing. The song's simple lyrics are as follows:

   ```
   99 bottles of beer on the wall, 99 bottles of beer.  Take one down, pass it around,
   98 bottles of beer on the wall.
   ```

   The same verse is repeated, each time with one fewer bottle. The song is completed when the singer or singers reach zero. Your task here is to write a Python program capable of generating all the verses of the song. Define a function `PrintVerse(NumberOfBottles)` to print all of the verses of the song up to and including `NumberOfBottles`.
2. Write an alternate function that takes a number of total bottles e.g. `FillVerses(Total)`, but instead of printing the verses, returns a list filled with each verse of the song. Use a list comprehension to fill the list, rather than for-loops. After calling the function, you should then be able to loop through the list and print the verses in the correct order.
   Hint: There are several ways in which to ensure that the verses are printed in order e.g. `reversed(), sorted(...,reverse=...)`

Exercise 2 - Powers

1. Can you complete the `RetSquared` function below by replacing the blank `<?>` with the correct variable name?

   ```
   def RetSquared(Number):
       # Returns the square
       Squared = <?> ** 2
       return(<?>)
   ```

2. Use the `RetSquared` function and a `for` loop to print the squares of all integers from 1 to 10.
3. Create a new function called `RetCubed` and use it to print cubes of all integers from 1 to 10.
4. Can you write a general function for raising numbers to powers? The function should take two arguments `Number` and `Power` and return the number raised to the given power? Remember to give your new function a sensible name.

5. Use your new general function to print the powers of 2 up to 2^{10}.
6. Change this function so that the argument `Power` has the default value `1`.
7. Write a new function which has arguments `Number` and `Powers`. If I have inputs `Number=2` and `Powers=[1, 3, 4]`, the function should return the list `[2, 8, 16]`.
   How else can you write a function when you don't know the exact number of inputs?
8. (*) Can you write a function which returns the prime factorization of a number?
   Note: Learn about prime factorization here: `https://www.mathsisfun.com/prime-factorization.html`

Exercise 3 - Word Scrambler

An old internet meme once claimed that if you scrambled the words of a sentence in such a way that the first and last letters remained in place, but the rest of the letters were shuffled, it could still be understood. The following sentence was used to promote the claim:

"Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe."

It was later proven incorrect; however, it makes for an interesting programming exercise!

How would we write a program which reads in a phrase and prints out the scrambled words? Let's break it down into steps.

1. First let's just consider how would we completely scramble a single word? For example, `"scramble"` could become `"lbeacmrs"` or `"toblerone"` could become `"loonberet"`.
   Hint: Recall the `shuffle` function in the `random` module. However, be aware that `shuffle` rearranges elements in place and so cannot work on strings - you will need to convert to a list.
2. So now we can scramble the entire word - how would you adapt this to keep the first and last letters in place? For example, `"scramble"` could become `"smlabcre"` or `"toblerone"` could become `"tnobleroe"`.
3. Now rewrite this as a function called `WordScrambler` which takes in an argument `Word`.
4. Almost there - can you apply your function to every word in a list of words? For example, the list `["scramble", "toblerone", "smell", "cheese"]`?
5. Finally, rather than have a list of words it'd be nicer to read in a sentence from the user using `input()` and print out the scrambled sentence. How would you do this?
6. (*) Enhance your word scrambler. Some ideas include:
   - Leave punctuation and numbers unchanged.
   - Add a flag to the function which switches functionality between scrambling the entire word and leaving the first and last letters in place.
   - How would you create a scrambler which only scrambles the position of every other letter?

Part 2. Functions arguments & Scope

Exercise 4 - Variable scope

Variable scope. It is important to understand a little bit about **scope** – an important concept in programming. The following exercises will demonstrate how variables are treated depending on whether they are declared inside (local scope) or outside (global scope) of a function.

1. Run the following code, predict the value printed to the screen.

```
def F():
    print(S)

S = "I hate spam"
F()
```

2. Run the following code, predict the values printed to the screen.

```
def F():
    S = "Me too"
    print(S)

S = "I hate spam"
F()
print(S)
```

3. Run the following code, predict the values printed to the screen.

```
def F():
    global S
    S = "Me too"
    print(S)

S = "I hate spam"
F()
print(S)
```

   Here we tell the interpreter that the variable `S` is `global` and so a local variable `S` is not being created by Python. Instead, S refers to the *global variable* `S`, or the variable S that was created in the main body of the program, and not in the function `F()`.
   Check that you understand how the scope of the variable impacts its use.

4. To further illustrate the point, try the following:

```
def F():
    S = "I am a variable"
    print(S)

F()
print(S)
```

   Because `S` is only created and used inside `F()`, and is not passed back to the main body of the program where `F()` is called, then `S` is not defined in the main body of the program.

Exercise 5 - Variadic functions and returning values.

Variadic functions can take any number of arguments, just like the print function, which works when used in the following way:

```
print("Hello, I am", Age, "years old, born in the month of", Month, ".")
```

1. Write a function that take an unknown amount of numbers (integers or floats) and then:
   - Calculates the sum of the numbers
   - Calculates the product of all of the numbers

   and **returns** these values as opposed to printing them from the function.
2. Write a function that takes an unknown amount of numbers as its arguments and returns the `min`, `max` and `average` of those numbers, *without* returning these as a list.

Part 3. Recursive functions

Exercise 6 - Recursive functions

1. Write a program that calculates the Fibonacci sequence using a recursive function called `Fib(n)` where `n` is the n^{th} number in the sequence.

   The following code should print:

   ```
   1  1  2  3  5  8  13  21  34  55  89  144  233  377
   ```

   ```
   for i in range(1,15):
       print(Fib(i), end=" ")
   ```

   If you are struggling, start by researching how to calculate the Fibonacci sequence. There are lots of tutorials online, so be sure to search for implementations of the Fibonacci function using *recursion*.

   Hint: You will need to identify the "special cases" of your program which prevent the function from calling itself infinitely many times.

2. Read about the relative advantages and disadvantages of using *recursive* functions i.e. what happens if `n >= 100`? Is there another version of the Fibonacci function that does not use recursion? Can this be called on large values of `n`?

Exercise 7 - (*) Additional Problem Ideas

If you finish all the exercises in this week's sheet, here are some additional problems:

1. Look back through previous exercise sheets and think about how you would solve them now using functions. Try some! I recommend:
   - The circles question at the end of Week 2 - Exercise 7
   - FizzBuzz Game from Week 3 - Exercise 8

2. Exercise **??** showed how dictionaries can be used to store information like databases and then queried using functions. Can you think of some other useful applications for this? For example, maybe a record of the books you have read and your review scores for them?
3. Explore different solutions for the Tower of Hanoi problem - description here: `https://en.wikipedia.org/wiki/Tower_of_Hanoi`
4. Hangman - you can find the rules here: `https://m.wikihow.com/Play-Hangman`
5. Blackjack - you can find the rules here: `https://entertainment.howstuffworks.com/how-to-play-blackjack.htm`