List comprehension - a quick reminder

1. Use dictionary comprehension to create a dictionary of numbers 1- 100 to their squares (i.e. {1:1, 2:4, 3:9 …}
2. Use list comprehension to create a list of prime numbers in the range 1-100

http://www.secnetix.de/olli/Python/list_comprehensions.hawk

Files

Use *open, read, write, close* which are explained in these tutorials:
https://www.tutorialspoint.com/python/python_files_io.htm
https://www.guru99.com/reading-and-writing-files-in-python.html
to:

1. Write "Hello world" to a file
2. Read the file back into python.
3. Write the square roots of the numbers 1-100 into the file, each in new line.

With

Read about with in one of the following links:
https://stackoverflow.com/questions/1369526/what-is-the-python-keyword-with-used-for
http://effbot.org/zone/python-with-statement.htm

1. Open a file using with and write the first paragraph from (you don't need to read the webpage, just copy paste it into code) https://en.wikipedia.org/wiki/Eigenface
2. Open the file using with and read the contents using readlines.

Strings

1. Split the paragraph to a list of words by spaces (hint: split())
2. Join the words back into a long string using "join". (hint: join())
   * (optional) Create a paragraph with the word order reversed in each sentence (but keep the order of the sentences)
3. write a function accepting two numbers and printing "the sum of __ and ___ is ___".
   Do this twice using one string formats learned in class each time.

Hint: https://pyformat.info/

Datetime, time, timeit

Read the current datetime to a variable usint datetime.datetime.no see:
https://docs.python.org/3/library/datetime.html

1. Print the date of the day before yesterday, and the datetime in 12 hours from now

Use *time* library to time library to time operations by storing time.time() before and after running an operation https://www.tutorialspoint.com/python/time_time.htm

1. Time one of the comprehension exercises you've performed and compare to a simple for implementation, to determine which one is faster

1. Now use *timeit* library to time your list comprehension operation, see: https://docs.python.org/2/library/timeit.html

Exceptions

Read about exceptions from:
http://www.pythonforbeginners.com/error-handling/exception-handling-in-python

1. Enhance the function you wrote which reads numbers (Strings 3.)
    a. In case of a string input writing an error message instead of raising an error
    b. Raise an error in case one of the input numbers is > 100
    c. Create a code which tries to read from the dictionary of squares created

Logger

Read about loggers from the example:
https://fangpenlin.com/posts/2012/08/26/good-logging-practice-in-python/
Create a logger in any of your programs and log various messages to a file using a FileHandler, and to a the console using StreamHandler
*Optional : try setting different levels (e.g. debug for console and info for file) and different formats and try different levels of logs in your code.

Regular expressions (re)

You can read about regex from the following sources:
Python documentation - https://docs.python.org/3/library/re.html
General links about RegEx - http://www.regular-expressions.info/tutorial.html
And - https://www.icewarp.com/support/online_help/203030104.htm
(The last is more brief and practical)
Try to understand the difference between search and match (and compile), and how to match the strings exactly.

1. To get the hang of it, try to do a few of the the exercises in: https://regexone.com/
2. To get a sense of regex in python, take a look at the first few exercises in: http://www.w3resource.com/python-exercises/re/

3*. Optional: Write a regular expression for identifying email addresses, find and compare to regular expression found online.

Combine everything!

Write a simple “range calculator” function which reads a file and writes the result into another file. You should log the operations, and how much time they took and handle problematic inputs (strings, wrong characters, etc.). Validate the input using regex.
The file will contain in each line two numbers denoting a range of numbers and they can only be integers, an operation denoted by a sign (+,*,-,/,**) and a third number which will be applied with the operation for each of the numbers in the range. Note all numbers and signs are separated by blanks (please don’t use .csv readers and use plain readers):
5 100 - 17

18 25 * 2.784
(First line: subtract 17 from all numbers between 5 and 100, Second line: multiply by 2.784 all numbers between 18 and 25)

The result of each range should be written into one line, separated with commas and with the precision of two digits after the decimal point.