**Exercise**

In a separate file, write a program to print “Hello World!” 10 times.

1. Create a new file with the .py extension using your favorite editor
2. Save your file
3. Open a CMD terminal window
4. Run your program using

c:> python Your\_file\_name\_here.py

and observe the output.

Could you write the same program using 1-line? Note you do not need any loops or anything beyond what we have just covered.

**Exercise**

1. Given the following list

data=[5,4,6,1,9,0,3,9,2,7,10,8,4,7,1,2,7,6,5,2,8,2,0,1,1,1,2,10,6,2]

compute the average of this data. Hint: you need the sum built-in function.

**Exercises**

1. Build a dictionary to with keys as integers 0,1,2,3,4,5 and the corresponding squares of these integers as values.

2. Build a reverse-lookup for the dictionary in part 1. That is, given the squared integer (i.e., the value of the above dictionary), find the corresponding square root from the dictionary (i.e., the key of the above dictionary).

**Exercises**

1. Given the following list from last time,
2. data=[5,4,6,1,9,0,3,9,2,7,10,8,4,7,1,2,7,6,5,2,8,2,0,1,1,1,2,10,6,2]

remove all the duplicated elements from this list. Your output should be [5,4,6,1,9,0,3,2,7,10,8]. Note that the order is preserved.

1. Write a function that computes the sum of all positive integers up to a specified value, n.
2. Write a function that determines whether or not a strictly positive integer is perfect. A positive integer is perfect if the sum of all its proper divisors is equal to itself. For example, 28 is perfect because its proper divisors are [1,2,4,7,14] and the sum([1,2,4,7,14]) = 28.
3. Generate all subsets of a set. Note there will be 2\*\*len(x) subsets of the set x. For example, the set [0,1,2] has eight subsets: {[],[0],[1],[2],[0,1],[1,2],[0,2],[0,1,2]}. Note that you don’t have to mind the ordering of each subset. Hint: use recursion. Hint: You probably want to use listinstead of set.
4. Generate all ordered sub-lists for a given list. For example, x = [0,1,2] has sub-lists

[[0],[1],[2],[1,2],[0,1],[0,2],[0,1,2]]

Hint: use slice.

1. Implement a reverse dictionary lookup. For example, given the following dictionary:
2. d = {i:(i%2) **for** i **in** range(10)}
3. {0: 0, 1: 1, 2: 0, 3: 1, 4: 0, 5: 1, 6: 0, 7: 1, 8: 0, 9: 1}

return a list of all the dictionary keys that map to 0 and 1, respectively. In this case,

{0: [0, 2, 4, 6, 8], 1: [1, 3, 5, 7, 9]}

Note that the output is a dictionary that has a list for each of the values of d.

1. Write a function to return True or False based on whether two intervals represented as tuples overlap. For example, (0,1) overlaps with (0,10) so the function should return True. Likewise, (1,2) does not overlap (3,4) so the function should return False .

**Exercise**

**Exercise**

1. Recall the data from last time
2. data=[5,4,6,1,9,0,3,9,2,7,10,8,4,7,1,2,7,6,5,2,8,2,0,1,1,1,2,10,6,2]

Write three columns to a file

data\_value, data\_value\*\*2, (data\_value+data\_value\*\*2)/3.

1. Download this [corpus of 10,000 common English words](https://raw.githubusercontent.com/first20hours/google-10000-english/master/google-10000-english-no-swears.txt) and do the following:
   * Compute the average length of the words.
   * What is the longest word?
   * What is the longest word that starts with *s*
   * What is the most common starting letter?
   * What is the most common ending letter?
2. Using the same file as above, create a new file that consists of each consecutive non-overlapping sequence of five lines merged into one line. Here’re the first 10 lines:

the of and to a

in for is on that

by this with i you

it not or be are

from at as your all

have new more an was

we will home can us

about if page my has

search free but our one

other do no information time

**Exercise**

1. Write a function that takes a single string character (i.e., ‘a’,’b’,’c’) as input and returns True or False if that character represents a valid integer.
2. Write a function to compute the area of the rectangle given the lengths of its sides, but you will only compute for rectangles, not squares, so raise an exception if otherwise.
3. Given a list of color hex-codes ['#FFAABB'], write a function to convert these into a list of RGB-tuples. For example, [(255,170,187)] corresponds to the example above. Here is the list of hex-codes to convert:
4. ['#FAEBD7', '#00FFFF', '#7FFFD4', '#F0FFFF', '#F5F5DC', '#FFE4C4',
5. '#000000', '#FFEBCD', '#0000FF', '#8A2BE2', '#A52A2A', '#DEB887',
6. '#5F9EA0', '#7FFF00', '#D2691E', '#FF7F50', '#6495ED', '#FFF8DC',
7. '#DC143C', '#00FFFF', '#00008B', '#133B63', '#104E8B', '#008B8B',

'#B8860B', '#A9A9A9', ]

**Exercise**

Given the following string

**>>>** data = """Mary had a little lamb

**...** its fleece was white as snow

**...** and everywhere that Mary went

**...** the lamb was sure to go"""

1. Compute the average length of the words in it.
2. Excluding repeated words, re-compute the average length of the words.

**Exercise**

What is inadvisable about the following loop?

**>>>** data = range(10)

**>>> for** i **in** data:

**...**  **if** i % 2: *# odd*

**...**  data.remove(i)

**>>>** print data

[0, 2, 4, 6, 8]

**Exercise**

Given a set of weights {1,3,9,27}, show that you can weigh any number between 1 and 40. In other words, using the set above and the addition and subtraction operations, construct any integer between 1 and 40 without re-using elements. For example, 4 = 1+1+1+1 is not acceptable.

For example,

8 = 9 - 1

10 = 1 + 9

Hint: see the [itertools](https://docs.python.org/2/library/itertools.html) module.

**Exercise**

1. Implement the [Sieve of Eratosthenes](https://en.wikipedia.org/wiki/Sieve_of_Eratosthenes). In other words, create function that can compute all prime numbers up to a given fixed integer.
2. Implement the [Bubble sort](https://en.wikipedia.org/wiki/Bubble_sort) as a Python function.
3. Implement a sliding window for an arbitrary input list. The function should take the window width and the window increment as inputs and should produce a sequence of overlapping lists from the input list. For example, given x=range(15), the following is the output given a window width of 5 and window increment of 2.
4. [[0, 1, 2, 3, 4],
5. [2, 3, 4, 5, 6],
6. [4, 5, 6, 7, 8],
7. [6, 7, 8, 9, 10],
8. [8, 9, 10, 11, 12],
9. [10, 11, 12, 13, 14]]
10. Given an integer, print the next smallest and next largest number that have the same number of 1 bits in their binary representation. Hint:

print '{0:03b}'.format(4)

will print out the binary representation of an integer for the requested number of bits (i.e., above is for 3-bits).

1. Using the random.randrange function, generate 100 random integers between 0 and 9. Count how many of these integers fit in the intervals [0,2], [3,7], [8,9].

**Exercise: Fibonacci numbers using generators**

The Fibonacci numbers are defined by the following recursion:  with initial values . Using generators, compute the first ten Fibonacci numbers, [1,1,2,3,5,8,13,21,34,55,89].