Software Development -2020
sheet 4

1. Let's say I give you a list saved in a variable: a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]. Write one line of Python that takes this list a and makes a new list that has only the even elements of this list in it.
2. Take two lists, say for example these two:

   a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]

   b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]

   and write a program that returns a list that contains only the elements that are common between the lists (without duplicates). Make sure your program works on two lists of different sizes.
3. Ask the user for a number and determine whether the number is prime or not. (For those who have forgotten, a prime number is a number that has no divisors.).

4. Write a program that takes a list of numbers (for example, a = [5, 10, 15, 20, 25]) and makes a new list of only the first and last elements of the given list.

5. Write a program that asks the user how many Fibonnaci numbers to generate and then generates them.

6. Write a password generator in Python. Be creative with how you generate passwords - strong passwords have a mix of lowercase letters, uppercase letters, numbers, and symbols. The passwords should be random, generating a new password every time the user asks for a new password. Include your code in a main method.

7. Concatenate two lists index-wise

   Given lists:
   list1 = ["M", "na", "i", "Ke"]
   list2 = ["y", "me", "s", "lly"]

   Expected output:
   ['My', 'name', 'is', 'Kelly']

8. Given a Python list. Turn every item of a list into its square root

Given a list:

```
aList = [1, 2, 3, 4, 5, 6, 7]
```

Expected output:

```
[1, 4, 9, 16, 25, 36, 49]
```

9. Given a two Python list. Iterate both lists simultaneously such that list1 should display item in original order and list2 in reverse order

Given a list:

```
list1 = [10, 20, 30, 40]
list2 = [100, 200, 300, 400]
```

Expected output:

```
10 400
20 300
30 200
40 100
```

10. Remove empty strings from the list of strings

Given a list:

```
list1 = ["Mike", "", "Emma", "Kelly", "", "Brad"]
```

Expected output:

```
["Mike", "Emma", "Kelly", "Brad"]
```

11. Add item 7000 after 6000 in the following Python List

Given a list:

```
list1 = [10, 20, [300, 400, [5000, 6000], 500], 30, 40]
```

Expected output:

```
[10, 20, [300, 400, [5000, 6000, 7000], 500], 30, 40]
```

12. Given a nested list extend it with adding sub list ["h", "i", "j"] in a such a way that it will look like the following list

Given List:
list1 = ["a", "b", ["c", ["d", "e", ["f", "g"], "k"], "l"], "m", "n"]
Sub List to be added = ["h", "i", "j"]

Expected output:
['a', 'b', ['c', ['d', 'e', ['f', 'g', **'h', 'i', 'j'**], 'k'], 'l'], 'm', 'n']

13. Given a Python list, find value 20 in the list, and if it is present, replace it with 200. Only update the first occurrence of a value.
Given a list:
list1 = [5, 10, 15, 20, 25, 50, 20]

Expected output:
list1 = [5, 10, 15, 200, 25, 50, 20]

14. Given a Python list, remove all occurrence of 20 from the list.
Given a list:
list1 = [5, 20, 15, 20, 25, 50, 20]

Expected output:
[5, 15, 25, 50]

15. Given a list of integers:

1. Build a list of factorials from 0! up to 5 !.
2. The same operation as in (a) but by using list inclusion.
3. A list of odd-numbered factorials up to 5 !, constructed using map and filter.
4. List inclusion does the same, replacing map and filter and making the lambda expression unnecessary.

16. Working with reduce:

a) Starting in Python 3.0, the reduce function is no longer built-in.
b) Import the add module so as not to create a function for adding two numbers.

c) Calculate the sum of integers not greater than 99.

d) Solving the same problem with the sum function, you no longer need to import the addition function.

17. Write and test a function to simulate the python's map?

18. Write and test a function to simulate the python's reduce?