INFO-H-600 - Python
TP 6 - Advanced Python constructions

Ex. 1. Read the following code:

```
def init_matrix(n, m):
    line = [0 for i in range(m)]
    matrix = [line for i in range(n)]
    return matrix

matrix = init_matrix(3, 3)
matrix[1][1] = 1
print(matrix)
```

- According to you, what will be printed ?
- Test the following code and compare it with your previous answer. If there is a difference, try to understand what caused the difference.
- If needed, and using still list comprehesion, modify the function `init_matrix(n,m)` such that it would provide the same results as showned here under:

```
matrix = init_matrix(3,3)
matrix[1][1] = 1
print(matrix)
[[0, 0, 0], [0, 1, 0], [0, 0, 0]]
```

Ex. 2. Using list comprehension, write down le list of integers lower than 100 which are multiple of 5 or 7 but not multiple of 5 and 7.

Ex. 3. Write a function `my_filter` that receives a list `lst` and a boolean function `f` and which returns a list composed of the elements of `lst` for which `f` returns `True`.

To test that `my_filter` is workging, you will have to define your own boolean function and pass it to `my_filter` as an argument. Please use a `lambda` function for this. For instance, the call:

```
my_filter(['hello', 666, 42, 'Thierry', 1.5], lambda x : isinstance(x, int))
```

must return:

```
[666, 42]
```

Ex. 4. Write a function that will sort a list of string according to the alphabetical order of the last letter of the string. For instance:

```
l = ['abc', 'cdfe', 'cba', 'awb']
my_sort(l)
print(l)     # ['cba', 'awb','abc', 'cdfe']
```

Ex. 5. The n^{th} number of the fibonacci sequence, F_n is defined as follows:

$$F_n = F_{n-1} + F_{n-2}$$

with $F_0 = 0$ and $F_1 = 1$. Write a generator that computes the fibonacci sequence.

Ex. 6. Write a generator that will yield the n first prime numbers.

Ex. 7. Every non prime number can be decomposed into a product of a sequence of prime numbers. For example, $45 = 3 * 3 * 5$. Write a generator `decompose` that will yield, for a given input number n, the sequence of prime numbers that it is composing.

For example:

```
l = list(decompose(4563))
print("*".join([str(prime) for prime in l])) # 3*3*3*13*13
```

INFO-H-600 - Programmation
TP 6 - Advanced Python constructions
Corrections

Solution to the exercise 1:

```python
def init_matrix(n, m):
    matrix = [[0 for i in range(m)] for i in range(n)]
    return matrix
```

Solution to the exercise 2:

```python
l = [i for i in range(100) if (((i%5 == 0) or (i%7 == 0)) and not (i%5 == 0 and i%7 == 0))]
```

Solution to the exercise 3:

```python
def my_filter(lst, f):
    return [x for x in lst if f(x)]
```

Solution to the exercise 4:

```python
def my_sort(l):
    l.sort(key=lambda x: x[-1])
```

Solution to the exercise 5:

```python
def fibonacci():
    """ A generator for creating the Fibonacci numbers """
    a, b = 0, 1
    while True:
        yield a
        a, b = b, a + b

f = fibonacci()

for x in f:
    if x > 100:
        break
    print(x, end=" ")
```

Solution to the exercise 6:

```python
import math

def prime(n):
    for i in range(2, int(math.sqrt(n)) + 1):
        if n % i == 0:
            return False
    return n >= 2

def primes(n):
    i = 0
    founds = 0
    while founds < n:
        if prime(i):
            founds += 1
            yield i
        i += 1

for i in primes(15):
    print(i)
```

Solution to the exercise 7:

```
import math

def is_prime(n):
    for i in range(2, int(math.sqrt(n)) + 1):
        if n % i == 0:
            return False
    return n >= 2

def primes():
    i = 0
    while True:
        if is_prime(i):
            yield i
        i += 1


def decompose(n):
    reminder = n
    for prime in primes():
        while reminder % prime == 0:
            yield prime
            reminder = reminder/prime
        if reminder == 1:
            return

l = list(decompose(4563))
print("*".join([str(prime) for prime in l]))
```