
**Lab09**

COMP 125  Programming with Python

## Recording Disclaimer


The synchronous sessions are recorded (audiovisual recordings). The students are not required to keep their cameras on during class.

The audiovisual recordings, presentations, readings and any other works offered as the course materials aim to support remote and online learning. They are only for the personal use of the students. Further use of course materials other than the personal and educational purposes as defined in this disclaimer, such as making copies, reproductions, replications, submission and sharing on different platforms including the digital ones or commercial usages are strictly prohibited and illegal.

The persons violating the above-mentioned prohibitions can be subject to the administrative, civil, and criminal sanctions under the Law on Higher Education Nr. 2547, the By-Law on Disciplinary Matters of Higher Education Students, the Law on Intellectual Property Nr. 5846, the Criminal Law Nr. 5237, the Law on Obligations Nr. 6098, and any other relevant legislation.

The academic expressions, views, and discussions in the course materials including the audio-visual recordings fall within the scope of the freedom of science and art.

# Q1: Implement the Fibonacci sequence by using lists

Implement the function **fibonacci(n)**, such that the function returns a list containing the first **n** numbers in the sequence.

Hint: You can assume that **n** > 1 and the sequence always starts with [1, 1]

Sample output

```
In [76]: fibonacci(10)
Out[76]: [1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
```

# Q2: Implement **prime** function

Implement the function **prime(n)**, such that the function returns a list containing all prime numbers upto and including **n**.

Hint: Remember the **is_prime** function from Hw 2

Hint: Start with a list [ 2 ] and append new prime numbers to this list, so that your list grows with the iterations (if we print out the iteration variable (**i**) and the updates to the **prime_list** during execution it should look like this:

```
In [129]: prime(10)
[2]
i= 3
[2, 3]
i= 4
i= 5
[2, 3, 5]
i= 6
i= 7
[2, 3, 5, 7]
i= 8
i= 9
i= 10
Out[129]: [2, 3, 5, 7]
```

Sample output

```
In [90]: prime(2)
Out[90]: [2]

In [91]: prime(20)
Out[91]: [2, 3, 5, 7, 11, 13, 17, 19]
```

# Q2: Implement **prime2** function (version 2)

- Any integer **i**, that is not a prime number, can be expressed as the product of prime numbers.
- Since we will store the prime numbers in a list, rather than iterating through all the numbers upto **i** as potential divisors, we can simply iterate through the list of prime numbers we already found.
- Also remember that you can stop checking, once you reach a prime number **p**, where $\mathbf{p^2 \geq n}$.*
- Implement the function **prime2(n)**, such that the function returns a list containing all prime numbers upto and including **n**.
- You can test the execution time of your functions with the following code. Test the two versions prime and prime2. Do you see an improvement? Hint: Try it with large n (e.g. 1000, 10000, 100000, etc.)

```
In [8]: import timeit
   ...: n=100
   ...: print(timeit.timeit("prime(n)", setup="from __main__ import prime,n", number=1))
```

* If n = r * q and n <= p*p (where p is a prime number), then r (or p) must be less than p, hence we have already checked for it.