Computational Thinking (CO2412): Tutorial – Calendar Week 41

1. Let us jointly review the most important features of Python syntax.[1]

2. The Jupyter notebook from this week's lecture has been extended[2] by a more elaborate performance measurement comparing the iterative and recursive store-and-recall implementations of the Fibonacci sequence. We will have a look at this together.

There, the seaborn library is used to visualize the data.[3] If your installation does not contain that library ("**import** seaborn" fails), download and install the package;[4] as a fallback solution, any other plotting tool can be used as well, based on the data output.

3. Design and implement **a)** an iterative algorithm and **b)** a recursive algorithm that determines the greatest element of a list; *e.g.*, if the list [3, -1, 0.5, 4, 0, 1] is given, the value 4 should be returned. **c)** Visualize the performance of the two implementations as a function of the length of the list, using lists consisting of random numbers[5] as an input.

Submit the solution to **no. 3** as a Jupyter Notebook ipynb file, if possible via Blackboard.

Submission deadline: 30th October 2021; discussion planned for 11th November 2021. Group work by up to four people is welcome.

[1] Python Software Foundation, *Python Tutorial*, https://docs.python.org/3/tutorial/, **2021**.

[2] http://home.bawue.de/~horsch/teaching/co2412/material/iterations-and-recursions.ipynb

[3] See the seaborn website, https://seaborn.pydata.org/, for a presentation of its features.

[4] https://anaconda.org/anaconda/seaborn

[5] There are many ways to do this; one would be to "**import** random" and then create a list with n elements using a statement such as "my_list = [random.random() **for** i **in** range(n)]".