Lab Sheet: Python Classes

This lab sheet covers the section on Python Fundamentals Classes from the course on *Industrial Programming* (F21SC): http://www.macs.hw.ac.uk/~hwloidl/Courses/F21SC#slides. These exercises should be implemented in Python 3. The tasks can be performed on your local machine (using either Linux or Windows) or on the Linux lab (EM 2.50) machines in the department (accessible through `x2go`).

Most exercises are available from the `gitlab-student.macs.hw.ac.uk` server, and these are marked as ⩔. Use this HOWTO do exercises with the gitlab server for an introduction how to do these exercises and how to use the unit testing in the continuous integration backend to get immediate feedback on your solution. Try this hello world exercise ⩔, alongside watching the HOWTO, to try this yourself.

Python Fundamentals (Week 8)

Complete the Python Fundamentals lab exercises, and in particular do these two exercises that you know from the C# part of the course:

(GCD) Implement Euclid's greatest common divisor algorithm as a function with two `int` parameters. Gitlab exercise GCD ⩔

(MatMult) Implement matrix multiplication as a function taking two 2-dimensional arrays (of any size) as arguments. Use the `numpy` library, and check the sample usage for this library at the end of this slideset. Gitlab exercise Matrix Multiplication ⩔

Python Classes (Week 9)

To get started on the lab exercises on Python classes, first do **OOP1–3** on the Python Tutor ↗ then move on to the Gitlab exercises, and focus on the Binary Search Tree that you know from the C# part of the course:

(OOP1,OOP2,OOP3) Go to the Python Online Tutor web page ↗, and work through the introductory object-oriented programming examples (OOP1, OOP2, OOP3). These are beginner's exercises.

(ContainerNoDup) Implement a container class with methods for `adding` an element, `extending` with a list and a check for `contains`, and then implement a sub-class for a container that avoids adding duplicates (with the same methods). Gitlab ⩔.

(BinaryTree) Implement the data structure of binary search trees, using classes, with operations for inserting and finding an element.