Lab Sheet: C# Objects and Classes

This lab sheet covers the section on C# Objects and Classes from the course on *Industrial Programming* (F21SC): http://www.macs.hw.ac.uk/~hwloidl/Courses/F21SC#slides. These tasks can be done in Visual Studio, either on the local machine, or via a remote desktop on one of the Windows lab machines (using the keyserver). See the Week0 Course Resources overview for details.

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 the continuous integration backend of gitlab to get immediate feedback on your solution. Try this hello world with test exercise 🦊 (version for VS 🦊), alongside watching the HOWTO, to try this yourself.

Only the bold-face exercises are mandatory. Others are recommended if you are fairly new to object-oriented programming or need to brush up on your background.

Getting Started

If you haven't done so already, watch the HOWTOs related to C# and VisualStudio listed in the previous LabSheet, also listed on the HOWTOs section of the main course information page ↗.

Sample sources and slides

- Relevant Slides on C# Objects and Classes ↗
- Gitlab repo with the sample sources for the `person` example 🦊

C# Objects and Classes (Week 2/3)

- Implement the bank account example as discussed in the C# slides on classes
- Complete the points example and implement access to the x- and y-fields, using direct access, public methods, and (automatic) properties, respectively. Gitlab sample sources 🦊
- These exercises use inheritance and overloading: Gitlab exercise with template testing code 🦊
  - Define some kind of class with a method `Area()`, that works on different shapes.
  - Define classes `Circle`, `Rectangle` and `Square`, and inherit from `Shape` to make sure that they have access to the `Area()` method.
  - In the main program, let the user decide which object to construct, and depending on the object ask for the parameters (radius, square-length, rectangle height and width), and calculate the overall and per-shape area and print it
  - Write a (polymorphic) function `TotalArea()` that takes an array of shapes and calculates the total area covered by all elements.
- Implement basic arithmetic on complex numbers using operator overloading. Gitlab repo 🦊
- **Implement the data structure of binary search trees with operations for inserting and finding an element.** Gitlab repo 🦊 Gitlab repo for VS 🦊

The bold-face exercises, **gcd, mat-mult, trees**, should be completed and pushed to gitlab by Week 3.