Programming Methods in Scientific Computing

Winter semester 2019/2020

Prof. Dr. Marc Alexander Schweitzer
Clelia Albrecht and Albert Ziegenhagel

UNIVERSITÄT BONN

Exercise sheet 0.

Welcome to our practical lab „Programming Methods in Scientific Computing“, where we equip you with the necessary skills and tools to write numerical software.
This sheet provides installation guidelines for the necessary tools as well as exercises to make you familiar with the workflow employed in this lecture.
If you encounter any problems during the installation process or in the exercises, please write an e-mail and ask for help.
Please note that you also have the possibility to use the provided student computers of the INS.

The Terminal

The terminal is a program that allows users to interact with their operating system's services. This is especially useful when running programs without graphical user interfaces (e.g. your own) and therefore the terminal is an invaluable tool for programmers. It is also called shell, command line or console. Strictly speaking, these terms are names for different parts and aspects of the program, but they are commonly used interchangeably.
You should invest some time to get used to handling the command line efficiently. The sources below should help you to get familiar with the most important commands of your preferred platform.

- Unix-like systems (including Linux and macOS):
  - `https://www.youtube.com/watch?v=3_vgwWTo1Vk`
  - `https://en.wikipedia.org/wiki/List_of_Unix_commands`
- Windows:
  - `https://www.youtube.com/watch?v=TUNNmVeyjW0`
  - `https://programminghistorian.org/en/lessons/intro-to-powershell`
  - `https://devblogs.microsoft.com/scripting/table-of-basic-powershell-commands/`

Exercise 1. (Terminal)

Make yourself familiar with your terminal. Make sure you know how to

a) create a new directory,

b) change directory (change into child and parent directories),

c) list all files in a directory,

d) remove single files and remove all files in a directory,

e) run a program

from your command line.

Text Editors

Source files and scripts are composed and edited using a text editor.
We specifically recommend to use VS Code. It can be downloaded for all platforms on their website `https://code.visualstudio.com/`, where you can also find installation guides and documentation. You are allowed to use any other text editor, but we will not offer editor specific support for any editor other than VS Code.

Exercise 2. (Text editors)

Give VS Code at least a try. Make yourself familiar with your text editor and its features.

Git and GitHub

Git is a version control system that tracks changes in source code and simplifies software development in teams. Many software companies use it to organize their workflow and keep track of development progress.
There are several websites and companies that provide hosting for version control using Git. In this practical lab we use GitHub (`https://github.com/`).
We use Git and GitHub classroom (`https://classroom.github.com/`) to hand in exercises, automatically test your solutions (and thus give you feedback on your programs) and facilitate the work in groups.

Install Git on your computer

- **Linux:** Open a terminal and install Git using your package manager. For Ubuntu, this is done by running

  ```
  sudo apt-get update
  sudo apt-get install git
  ```

  For other Linux distributions, choose your corresponding package manager.

- **Windows:** We will use `chocolatey` to install most software on Windows.

  To install `chocolatey`, open an elevated (Administrator) PowerShell and run

  ```
  Set-ExecutionPolicy Bypass -Scope Process -Force; iex ((
     ↪ New-Object System.Net.WebClient).DownloadString('https
     ↪ ://chocolatey.org/install.ps1'))
  ```

  More information (including the command line above) can be found at

  `https://chocolatey.org/install#install-with-powershellexe`.

  After `chocolatey` has been installed successfully you can use

  ```
  choco install git -y
  ```

  to install Git.

- **macOS:** We will use `homebrew` to install most software on macOS.

  To install `homebrew`, open a terminal and run

  ```
  xcode-select --install
  /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent
     ↪ .com/Homebrew/install/master/install)"
  ```

  More information (including the command above) can be found at

  `https://brew.sh`.

  After `homebrew` has been installed successfully you can use

  ```
  brew install git
  ```

  to install Git.

Get familiar with GitHub

Here are some tutorials for using Git:

- `https://www.youtube.com/watch?v=5OLUolv4DaM&feature=youtu.be` (a well-structured demonstration of the Git workflow under Linux)
- `https://rogerdudler.github.io/git-guide/index.de.html`
- `https://guides.github.com/activities/hello-world/`

Exercise 3. (Start using Git and GitHub)

a) Install Git on your computer

b) Get an account on GitHub (`https://github.com/join?source=header-home`) and watch or read the tutorials above.

c) Get an account on GitHub Classroom (`https://classroom.github.com/`)

Python interpreter

The first programming language of this lecture is `Python 3` (`https://www.python.org/`). To run `Python` programs on your computer, you need a so-called `Python interpreter` (see instruction guidelines below). `Python` can be used both interactively in the interpreter as well as by executing `Python` scripts, i.e. text files with the ending `.py`. Note that your hand-in exercises will need to be `Python` scripts.
To install `Python 3` on your computer, follow these instructions:

- **Linux:** Open a terminal and run

  ```
  sudo apt-get update
  sudo apt-get install python3
  ```

  for Ubuntu. For other Linux distributions, choose your corresponding package manager.

- **Windows:** Open an elevated (Administrator) PowerShell and run

  ```
  choco install python -y
  ```

  to install `Python 3`.

- **macOS:** Open a terminal and run

  ```
  brew install python
  ```

  to install `Python 3`.

Exercise 4. (Python interpreter)

Install an `Python 3` interpreter on your computer and check that it installed successfully by running

```
python3 --version
```

Compiler for C/ C++

In this lab, we will also discuss the efficient implementation of numerical algorithms using the hardware-near programming languages `C` and `C++`. `C` and `C++` source code (text files with the endings `.c` and `.cpp`, respectively) is translated into executable programs using a so-called compiler. For this lab, we would recommend GCC/G++ for Linux, Microsoft Visual `C++` for Windows and `clang` for macOS, which can be installed following the instructions below.

- **Linux:** Open a terminal and run

  ```
  sudo apt-get update
  sudo apt-get install build-essential
  ```

  for Ubuntu. For other Linux distributions, choose your corresponding package manager. Check that it installed successfully by running

  ```
  gcc --version
  ```

- **Windows:** Open an elevated (Administrator) PowerShell and run

  ```
  choco install visualstudio2019buildtools --
  ↪ package-parameters "--add Microsoft.VisualStudio.
  ↪ Workload.VCTools --includeRecommended --quiet --locale
  ↪  en-US" -y
  ```

  This will probably take a while to finish. To confirm that the installation was successful you should run a `Developer PowerShell for VS2019` from the start menu and enter

  ```
  cl
  ```

  which should give you an output similar to the following:

  ```
  Microsoft (R) C/C++ Optimizing Compiler Version 19.22.27905 for x86
  Copyright (C) Microsoft Corporation.  All rights reserved.

  usage: cl [ option... ] filename... [ /link linkoption... ]
  ```

- **macOS:** `C` and `C++` compilers have been installed by

  ```
  xcode-select --install
  ```

  already. To check the compiler you can run

  ```
  clang --version
  ```

  which should give you an output similar to the following:

  ```
  Apple LLVM version 10.0.1 (clang-1001.0.46.4)
  arget: x86_64-apple-darwin18.6.0
  Thread model: posix
  InstalledDir: /Applications/Xcode.app/Contents/Developer/Toolchains/XcodeD
  ```

Exercise 5. (`C` and `C++` Compiler)

Install a `C` and `C++` compiler on your computer and check that it installed successfully.

Test Lab

To get you used to the workflow of this practical lab, here is a small test classroom assignment.

Exercise 6. (Test classroom assignment)

a) Accept the assignment on GitHub classroom from `https://classroom.github.com/a/fxO2Ktcq`. This will create a repository for your submissions that only you and the assistants of the course have access to.

b) Clone this repository to your local computer.

c) Start a new branch to work on.

d) Solve the exercise according to the information in the readme of the repository.

e) Stage, commit and push your changes.

f) Create a pull request on GitHub.

g) Repeat, if necessary until all tests pass.