Scientific Programming
(Wissenschaftliches Programmieren)

Exercise 11

1. Equation solver with arguments

- Change the script for solving the linear system of equations, so that it accepts an optional argument specifying the directory where the input for the equation can be found (and where the output will be written into).
- Test, whether you can run it with an input in a different directory as the current one.
- The help text (obtained with `./linsolver -h`) should look like the example given at the end of the exercises.
- Commit your changes.

2. Using a Python package

- Place the two modules of the linsolver project into a package called `linsolver` in order to avoid eventual name-conflicts with other Python-modules. Update the `import` statements accordingly. (You might also rename your module `linsolverio.py` to `io.py` now.)
- Create a __init__.py file, which imports the most important functions of your project directly into the `linsolver` name space, so that users of your project can access them without having to know, in which module they are.
- Commit your changes.

3. Cleaning up the project

- Check which entitites in your modules should be public. Make sure, that all other entities are marked as private (by using the "_" prefix).
- Make sure that each source file follows PEP 8. You might consider to reformat all source files with the `black` formatter.
- Make sure that each source file in your project has a high `pylint` score (at least 9.0).
- Make sure that `mypy` does not report any issues with your source files.
- Make sure that your tests are working and are extensive enough (check coverage).
- Make sure that the API-documentation generation via Sphinx works and produces the right documentation.
- Extend the `README.rst` file, so that all information necessary to create an input for your program, to run your program and to interprate the results are contained.
- Commit your changes.

4. Creating an installable Python package from your project

- Transform your project into a installable / distributable package with the package name `linsolver`.
- Make sure, that your projects layout aligns either with the flat-layout or with the src-layout.
- Test the package build by creating a Python wheel from your project.
- Test the package installation by installing the generated wheel into a separate virtual environment.
- Bump your projects version number to 1.0 (e.g. in `pyproject.toml`).
- Commit your changes and tag your commit with the tag **v1.0**.

Congratulations to the first stable release of your Python project!

Help page of linsolver (obtained via `linsolver -h`):

```
usage: linsolver [-h] [-d DIRECTORY]

Solves the linear sytem of equation Ax = b. It requires an input file
'linsolve.in', containing the coefficient matrix A (each row of A should be
written into a separate line) followed by the right-hand-side vector b in a
single row. The script writes a file 'linsolve.out' containing the solution
vector x or emits an error message and exits with a non-zero exit code, if the
linear system of equations could not be solved.

optional arguments:
  -h, --help            show this help message and exit
  -d DIRECTORY, --directory DIRECTORY
                        directory, where input file is located and where
                        output should be written to (default: .)
```