Scientific Programming
(Wissenschaftliches Programmieren)

Exercise 8

1. Gaussian elimination with dependency detection

- Extend the Gaussian elimination with detection for linear dependency.
- The program should raise an appropriate exception (e.g. ValueError) if the system of equations is linearly dependent.
- Modify the test with the linearly dependent system of equation in `test_solvers.py` to check whether the correct exception is raised.
- Commit your changes.

2. Gaussian elimination as a program

- Create an executable script, which reads a linear system of equations from a file `linsolve.in`, solves the linear system of equation and writes the results into a file `linsolve.out`.
- The script should import the solver module (solvers.py) and use the routines therein to solve the linear system of equation.
- The script should do its file I/O using routines from a special module (linsolveio.py):
  - A routine which reads the input from a given file (read_input)
  - A routine which writes the result into a given file (write_result)
  - A routine which writes an error message in a file (write_error)
- The input file should have following format:

```
Matrix_A_1st_line
Matrix_A_2nd_line
…
RHS_of equation_as_one_line
```

  Example:

```
2.0  4.0  4.0
1.0  2.0 -1.0
5.0  4.0  2.0
1.0  2.0  4.0
```

- The output file should contain the solution vector in one row, or the string "`ERROR::LINDEP: Linearly dependent equations`" if the linear system of equation is linearly dependent.

  Example output when linear system of equation has a solution:

```
0.666666666666667 0.416666666666667 -0.5
```

- Example output when linear system of equation is linearly dependent

  ```
  ERROR::LINDEP: Linearly dependent equations
  ```

- Make sure, the `linsolve` script exists with a well formatted error message gracefully (and not with an unhandled exception), when the input file is not found.
- You can use the `numpy.loadtxt()` and the `numpy.savetxt()` routines to read/write the arrays.
- Commit your changes to the repository.

3. Parameterized tests

- Rewrite the parametrized tests in `test_solvers.py`, so that they read the data for A, b and the expected solution x using the routines from the exercise above.
- Commit your changes (including the test data in the `testdata/` folder)

4. API-documentation

- Set-up in the `docs/` subfolder of your project a Sphinx-project, which automatically extracts the API-documentation from your sources.
- Check that the API-documentation is created correctly.
- Add the relevant files to your repository (and the irrelevant files/directories to the `.gitignore` file).

5. LU-decomposition*

Rewrite the Gaussian-elimination algorithm into an LU-decomposition. It should contain the routines `ludecompose()` and `lusolve()` for making the LU-decomposition of the matrix A and for solving the equation with the help of the LU-decomposed form, respectively.