# Problem Sheet:
# Code Quality for Software

## Part 1: Debugging warm-up questions.

1. Tasks debug_{1..5}.py in your practical assignment.

## Part 2: Debugging with pdb tools

1. debug_pdb_1.py
   Debug the following implementation on the Selection Sort algorithm using pdb.

## Part 3. Profiling with CProfile.

1. profiling_1.py
   Place the following functions below in order of their efficiency. They all take in a list of numbers between 0 and 1. The list can be quite long. An example input list would be [random.random() for i in range(1000000)]. How would you prove that your answer is correct?

2. profiling_2.py
   Find the performance bottleneck in the provided code and optimize it. Demonstrate the improved performance by comparing the original and optimized code.

3. Plot Profiles of selection sort using snakeviz. Test it on 100 samples and 10000 samples. Which of these algorithms is preferable in terms of speed?

## Part 4. Code linting

1. linting_1.py
   The goal of this task is to bring this file to full PEP8 compatibility.
   Recommended actions:

   a. Install and run pylint on the source code to perform an initial assessment of the code quality and PEP8 compatibility.

   b. Install and run a reformatting tool (e..g Black) to update the code automatically, then run pylint again to see if there is any improvement.

   c. Continue with manual reformatting to further improve the code, periodically checking your progress with pylint.

2. linting_2.py
   The goal of this task is to assess and reduce the complexity of the Python program. Use Radon package's model of Cyclomatic complexity (`radon cc`) to improve this program.