3rd programming assignment

1. Implement feedforward neural networks with a fixed architecture

- Submission
  - Source code filling the missing parts in the attached python file `fnn.py`
    - `fnn.py` implements a FNN comprising two hidden (fully connected) layers
    - The missing parts are marked by comment ”TODO: IMPLEMENT”
  - Submitted code is required to pass the test in `fnn.py`

3rd programming assignment

1 (optional). Implement feedforward neural networks with a more flexible architecture

- Submission
  - Source code implementing FNNs where:
    - They may have any number of hidden layers
    - Each hidden layer may have any number of neurons
    - Activation functions for hidden layers may be selected from ones you implemented
      - You must implement one or more additional activation functions
      - Examples of activation functions: tanh and ReLU
      - See https://en.wikipedia.org/wiki/Activation_function for detail
  - Submitted code is required to pass the same test as `fnn.py` (by fixing an architecture configuration)
- Remark: it is fine to extend `fnn.py` for this exercise