Exercise Sheet #5

Problem 1 (*Implementing a Feedforward network*)

According to Richard Feynman, you only really understands something if you know multiple ways to do it. In his spirit, let's implement a simple fully connected feedforward network with two hidden layers in various different ways in PyTorch. The hidden layers perform a linear transformation and apply a ReLU activation function. Consider the example from the lecture (link). Define the model

(a) by explicitly defining the module parameters in the constructor of your `nn.Module` subclass and implementing the necessary matrix multiplications in the `forward` function.

(b) using `nn.Linear` and `nn.ReLU`.

(c) using a `nn.ModuleList`.

(d) using `nn.Sequential`.

Problem 2 (*Digging into the Code*)

In this directory (link), you can find the implementations of all available PyTorch modules. With your current knowledge of ML and Python, look into

(a) how the `nn.Module` base class is implemented.

(b) how the `nn.Linear` module is implemented.

(c) how certain functionalities work (e.g. passing input to a layer directly to the class instance[1] or how the `parameters()` method works).

(d) how the `nn.ReLU` module works.

[1]Hint: For this a `__call__` method must be implement