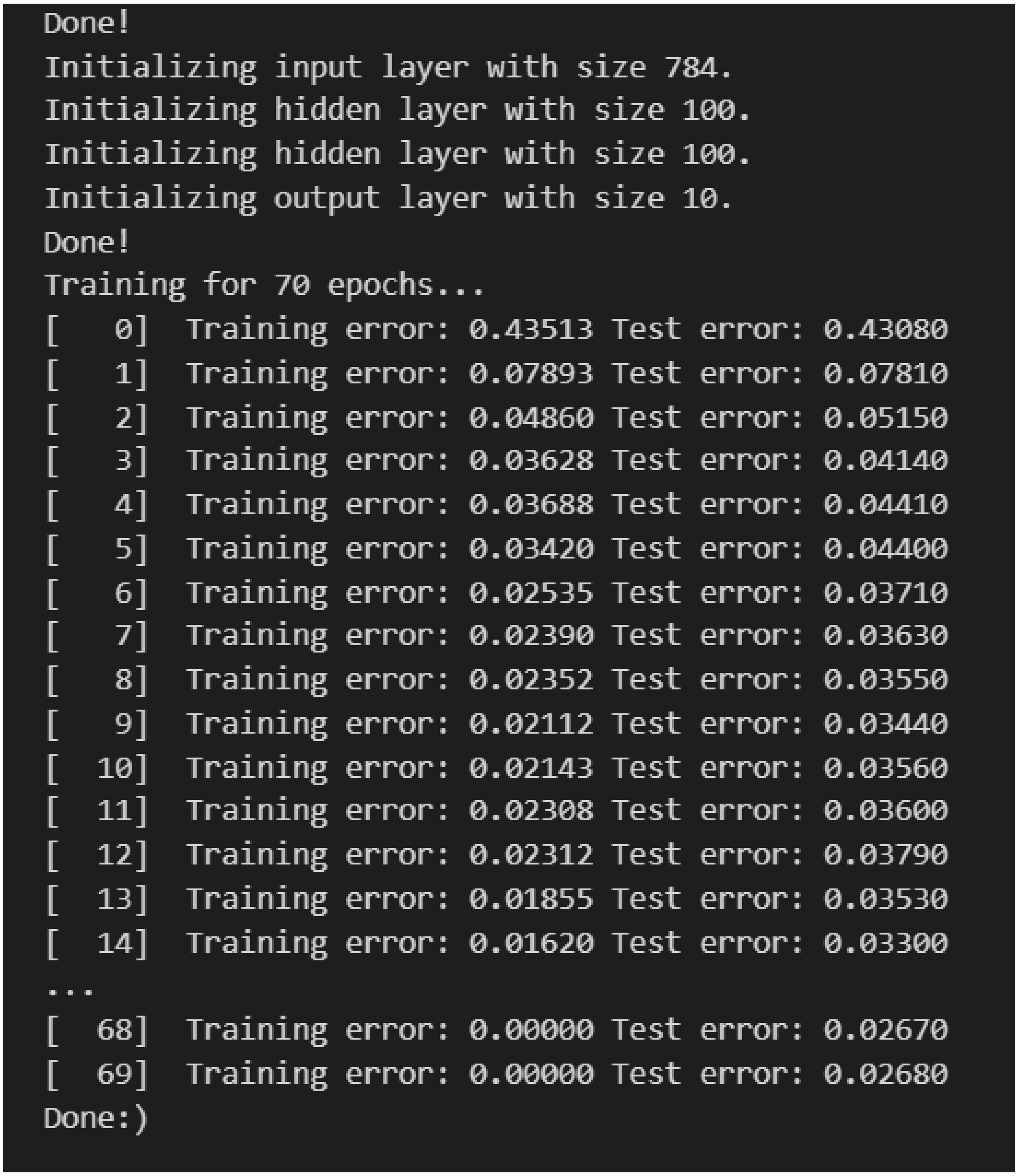
1. Understand the implementation structure of the multilayer perceptron in Jupyter notebook Lab5_ANN_backprop.

a. Be able to explain the principle of backpropagation algorithm;
b. Be able to explain the meaning and the role of the Softmax function;
c. Be able to name typically used non-linear output functions and implications of choosing one or another for implementation.
d. The code in the provided Jupyter notebook will stop execution at several points. Find the places in the code, where the execution breaks, answer the questions, comment out the exit line and run the code again.

2. Run the code with the suggested configuration of the hyperparameters: number of epochs = 70 and learning rate =0.05. What is the classification accuracy?



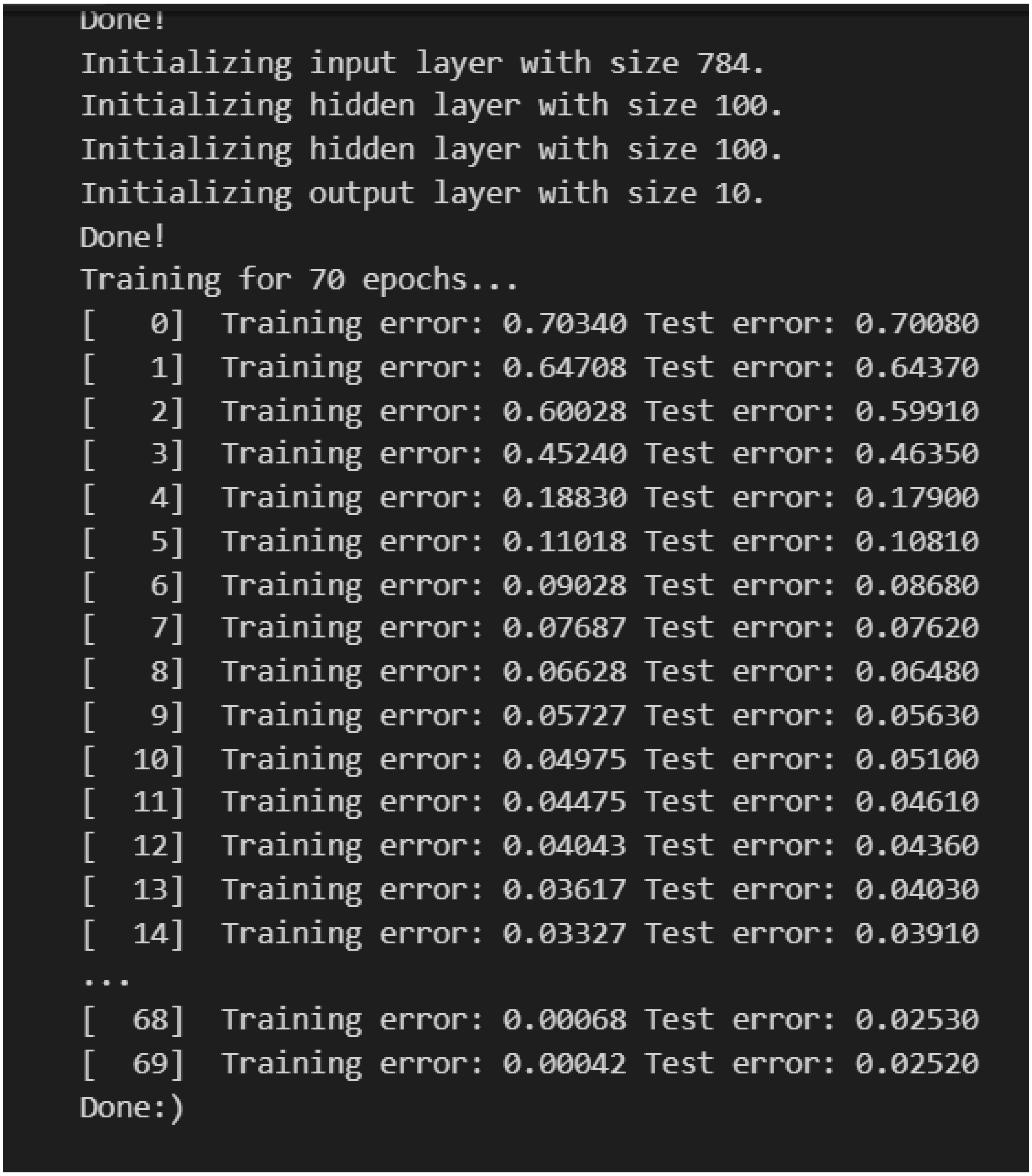
```
Done!
Initializing input layer with size 784.
Initializing hidden layer with size 100.
Initializing hidden layer with size 100.
Initializing output layer with size 10.
Done!
Training for 70 epochs...
[   0]  Training error: 0.43513 Test error: 0.43080
[   1]  Training error: 0.07893 Test error: 0.07810
[   2]  Training error: 0.04860 Test error: 0.05150
[   3]  Training error: 0.03628 Test error: 0.04140
[   4]  Training error: 0.03688 Test error: 0.04410
[   5]  Training error: 0.03420 Test error: 0.04400
[   6]  Training error: 0.02535 Test error: 0.03710
[   7]  Training error: 0.02390 Test error: 0.03630
[   8]  Training error: 0.02352 Test error: 0.03550
[   9]  Training error: 0.02112 Test error: 0.03440
[  10]  Training error: 0.02143 Test error: 0.03560
[  11]  Training error: 0.02308 Test error: 0.03600
[  12]  Training error: 0.02312 Test error: 0.03790
[  13]  Training error: 0.01855 Test error: 0.03530
[  14]  Training error: 0.01620 Test error: 0.03300
...
[  68]  Training error: 0.00000 Test error: 0.02670
[  69]  Training error: 0.00000 Test error: 0.02680
Done:)
```

This was: Accuracy: 0.9732

3. Run the code with Learning rate =0.005 and Learning rate =0.5. Explain the observed differences in the functionality of the multi-layer perceptron.

<u>0.005</u>

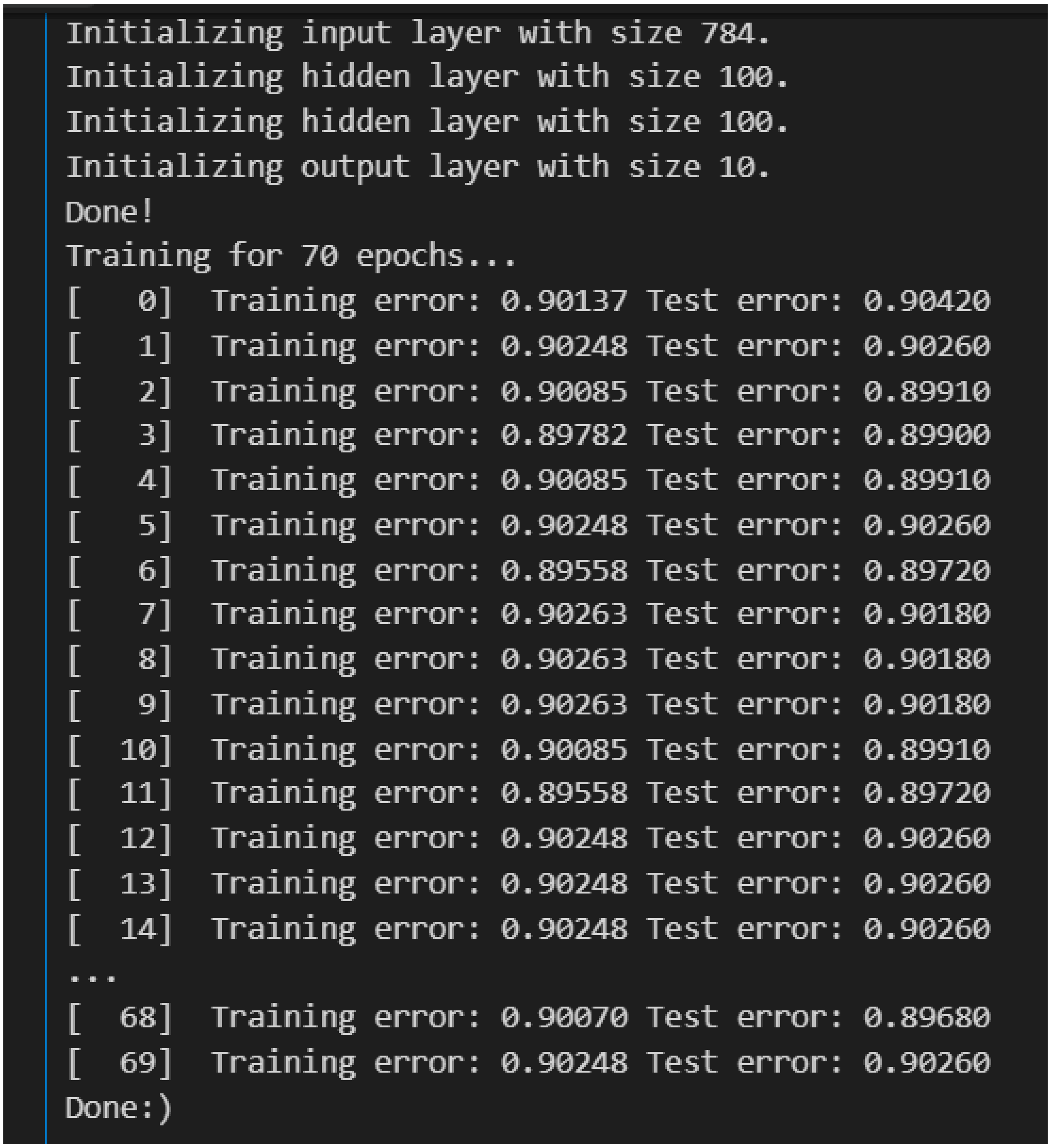
Compared to 0.05 it might of taken longer for each epoch but I did remove my charger which puts my computer into power saving mode and decrease performance output.



```
Done!
Initializing input layer with size 784.
Initializing hidden layer with size 100.
Initializing hidden layer with size 100.
Initializing output layer with size 10.
Done!
Training for 70 epochs...
[   0]  Training error: 0.70340 Test error: 0.70080
[   1]  Training error: 0.64708 Test error: 0.64370
[   2]  Training error: 0.60028 Test error: 0.59910
[   3]  Training error: 0.45240 Test error: 0.46350
[   4]  Training error: 0.18830 Test error: 0.17900
[   5]  Training error: 0.11018 Test error: 0.10810
[   6]  Training error: 0.09028 Test error: 0.08680
[   7]  Training error: 0.07687 Test error: 0.07620
[   8]  Training error: 0.06628 Test error: 0.06480
[   9]  Training error: 0.05727 Test error: 0.05630
[  10]  Training error: 0.04975 Test error: 0.05100
[  11]  Training error: 0.04475 Test error: 0.04610
[  12]  Training error: 0.04043 Test error: 0.04360
[  13]  Training error: 0.03617 Test error: 0.04030
[  14]  Training error: 0.03327 Test error: 0.03910
...
[  68]  Training error: 0.00068 Test error: 0.02530
[  69]  Training error: 0.00042 Test error: 0.02520
Done:)
```

This was: Accuracy: 0.9748

0.5

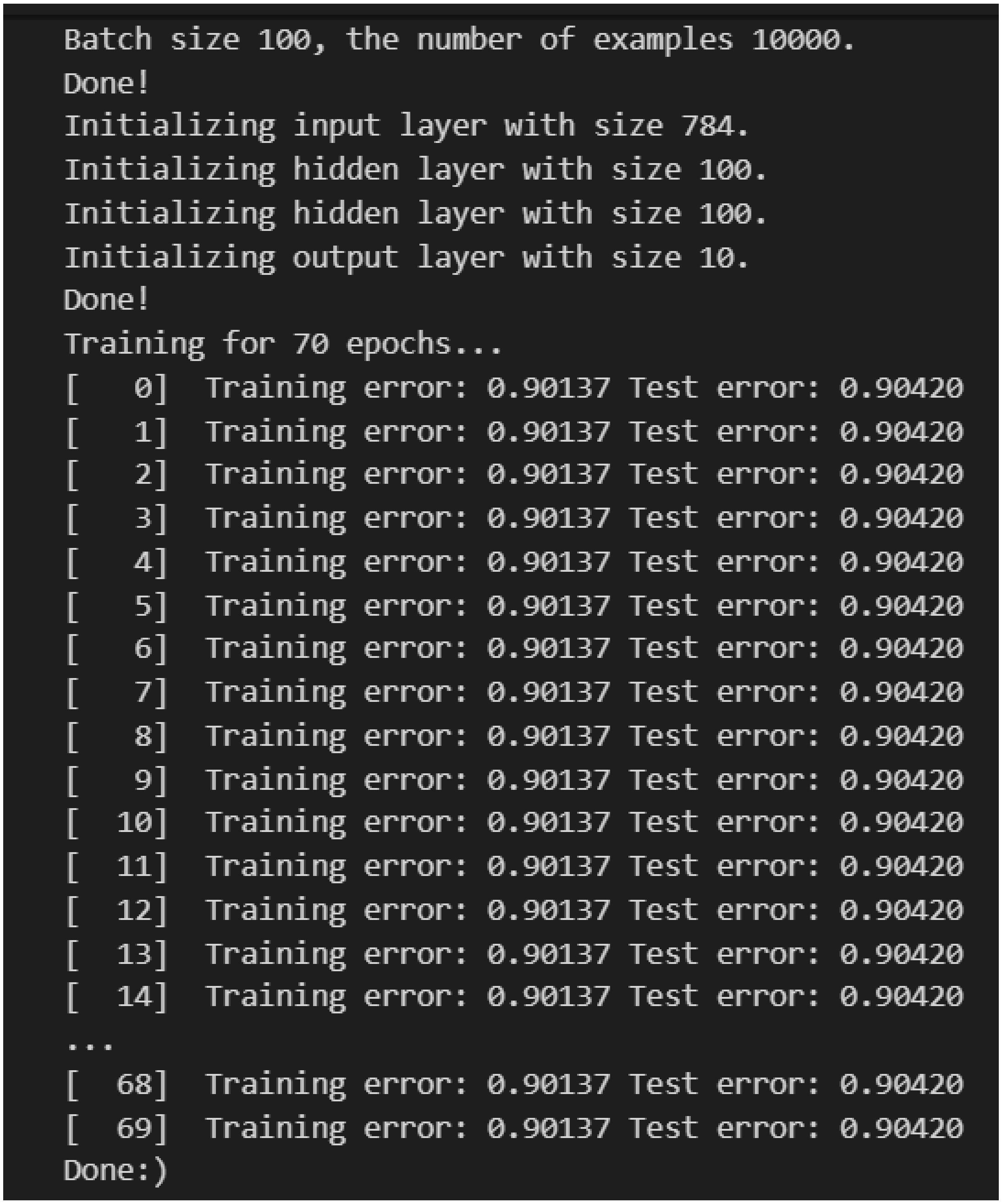


```
Initializing input layer with size 784.
Initializing hidden layer with size 100.
Initializing hidden layer with size 100.
Initializing output layer with size 10.
Done!
Training for 70 epochs...
[   0]  Training error: 0.90137 Test error: 0.90420
[   1]  Training error: 0.90248 Test error: 0.90260
[   2]  Training error: 0.90085 Test error: 0.89910
[   3]  Training error: 0.89782 Test error: 0.89900
[   4]  Training error: 0.90085 Test error: 0.89910
[   5]  Training error: 0.90248 Test error: 0.90260
[   6]  Training error: 0.89558 Test error: 0.89720
[   7]  Training error: 0.90263 Test error: 0.90180
[   8]  Training error: 0.90263 Test error: 0.90180
[   9]  Training error: 0.90263 Test error: 0.90180
[  10]  Training error: 0.90085 Test error: 0.89910
[  11]  Training error: 0.89558 Test error: 0.89720
[  12]  Training error: 0.90248 Test error: 0.90260
[  13]  Training error: 0.90248 Test error: 0.90260
[  14]  Training error: 0.90248 Test error: 0.90260
...
[  68]  Training error: 0.90070 Test error: 0.89680
[  69]  Training error: 0.90248 Test error: 0.90260
Done:)
```

This was: Accuracy: 0.0974

Summary: It is observed that the higher the learning rate the worse the accuracy and larger the error gets.

4. Extend the code implementing the ReLU output function. Run the perceptron with the suggested by default configuration of hyperparameters: number of epochs = 70 and learning rate =0.05. What is the classification accuracy?



```
Batch size 100, the number of examples 10000.
Done!
Initializing input layer with size 784.
Initializing hidden layer with size 100.
Initializing hidden layer with size 100.
Initializing output layer with size 10.
Done!
Training for 70 epochs...
[   0]  Training error: 0.90137 Test error: 0.90420
[   1]  Training error: 0.90137 Test error: 0.90420
[   2]  Training error: 0.90137 Test error: 0.90420
[   3]  Training error: 0.90137 Test error: 0.90420
[   4]  Training error: 0.90137 Test error: 0.90420
[   5]  Training error: 0.90137 Test error: 0.90420
[   6]  Training error: 0.90137 Test error: 0.90420
[   7]  Training error: 0.90137 Test error: 0.90420
[   8]  Training error: 0.90137 Test error: 0.90420
[   9]  Training error: 0.90137 Test error: 0.90420
[  10]  Training error: 0.90137 Test error: 0.90420
[  11]  Training error: 0.90137 Test error: 0.90420
[  12]  Training error: 0.90137 Test error: 0.90420
[  13]  Training error: 0.90137 Test error: 0.90420
[  14]  Training error: 0.90137 Test error: 0.90420
...
[  68]  Training error: 0.90137 Test error: 0.90420
[  69]  Training error: 0.90137 Test error: 0.90420
Done:)
```

This was: Accuracy: 0.0958