Grupo

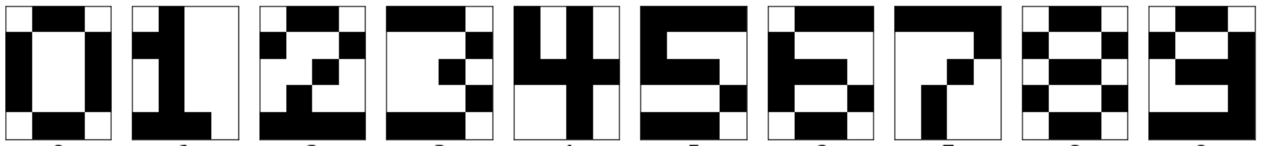
Conrado Luiz, Luíza Guedes, Marcelo Barros, Igor Feital

In [1]:
```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import numpy as np
from IPython.display import display
from matplotlib import rcParams
```

In [2]:
```
images = {}

for i in range(10):
    images[i] = mpimg.imread(f'data/{i}.png')

images[0]
```

Out[2]:
```
array([[[1., 1., 1., 1.],
        [0., 0., 0., 1.],
        [0., 0., 0., 1.],
        [1., 1., 1., 1.]],

       [[0., 0., 0., 1.],
        [1., 1., 1., 1.],
        [1., 1., 1., 1.],
        [0., 0., 0., 1.]],

       [[0., 0., 0., 1.],
        [1., 1., 1., 1.],
        [1., 1., 1., 1.],
        [0., 0., 0., 1.]],

       [[0., 0., 0., 1.],
        [1., 1., 1., 1.],
        [1., 1., 1., 1.],
        [0., 0., 0., 1.]],

       [[1., 1., 1., 1.],
        [0., 0., 0., 1.],
        [0., 0., 0., 1.],
        [1., 1., 1., 1.]]], dtype=float32)
```

In [3]:
```
rcParams['figure.figsize'] = 20 ,15

fig, ax = plt.subplots(1, 10)


for axis, (i, pixels) in zip(ax.reshape(-1), images.items()):
    axis.imshow(pixels)
    axis.set_xticks([])
    axis.set_yticks([])
    axis.set_xlabel(i, fontsize=20)
```



In [4]:
```
numbers = {}

for i, pixels in images.items():
    number = []
    for row in pixels:
        for col in row:
            number.append(int(not col[0]))
    numbers[i] = number

numbers
```

Out[4]:
```
{0: [0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0],
 1: [0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0],
 2: [0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1],
 3: [1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0],
 4: [1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0],
 5: [1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0],
 6: [0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0],
 7: [1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1],
 8: [0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0],
 9: [0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1]}
```

In [5]:
```
def onehot(n_classes, index):
    arr = [0] * n_classes
    arr[index] = 1
    return arr

onehot(10, 0)
```

Out[5]:
```
[1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

In [6]:
```
def make_xy(_dict):
    x = []
    y = []

    for key, values in _dict.items():
        x.append(values)
        y.append(onehot(10, key))

    x = np.array(x)
    y = np.array(y)

    return x, y

x, y_true = make_xy(numbers)

print(x)
print('\n', y_true)
```

```
[[0 1 1 0 1 0 0 1 1 0 0 1 1 0 0 1 0 1 1 0]
 [0 1 0 0 1 1 0 0 0 1 0 0 0 1 0 0 1 1 1 0]
 [0 1 1 0 1 0 0 1 0 0 1 0 0 1 0 0 1 1 1 1]
 [1 1 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 0]
 [1 0 1 0 1 0 1 0 1 1 1 1 0 0 1 0 0 0 1 0]
 [1 1 1 1 1 0 0 0 1 1 1 0 0 0 0 1 1 1 1 0]
 [0 1 1 1 1 0 0 0 1 1 1 0 1 0 0 1 0 1 1 0]
 [1 1 1 1 0 0 0 1 0 0 1 1 0 1 0 0 0 1 0 1]
 [0 1 1 0 1 0 0 1 0 1 1 0 1 0 0 1 0 1 1 0]
 [0 1 1 0 1 0 0 1 0 1 1 1 0 0 0 1 1 1 1 1]]

[[1 0 0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0 0 0]
 [0 0 0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0 0 1]]
```

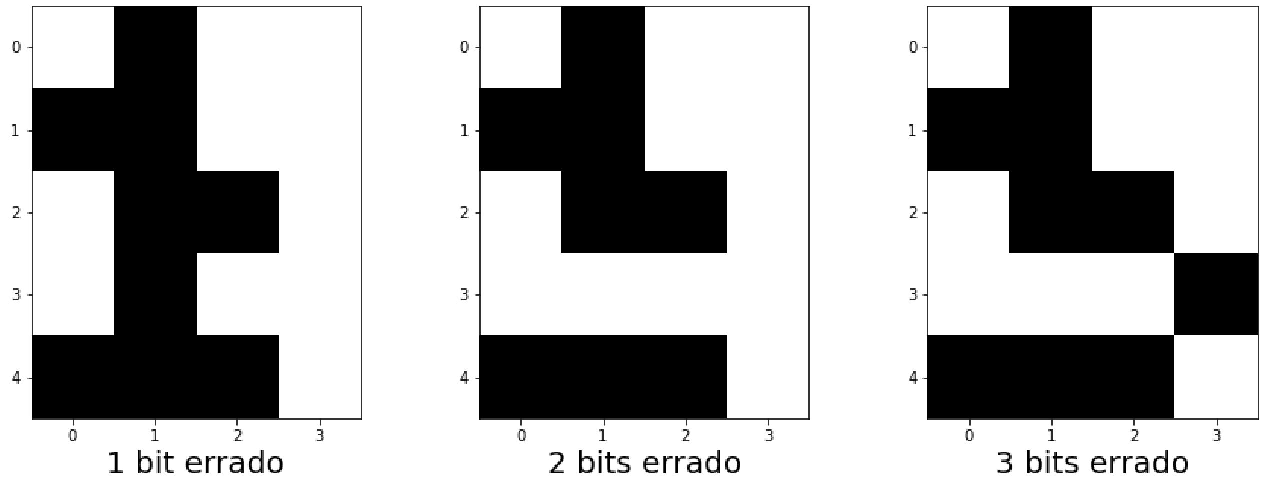
In [7]:

```
one_wrong_set = {}
two_wrong_set = {}
three_wrong_set = {}

np.random.seed(6)

for key, value in numbers.items():
    random = np.random.randint(0, 20, size=3)

    # ----------------------------------------------------------------
    new_value = value.copy()
    new_value[random[0]] = int(not new_value[random[0]])

    one_wrong_set[key] = new_value

    # ----------------------------------------------------------------
    new_value = value.copy()
    new_value[random[0]] = int(not new_value[random[0]])
    new_value[random[1]] = int(not new_value[random[1]])

    two_wrong_set[key] = new_value

    # ----------------------------------------------------------------
    new_value = value.copy()
    new_value[random[0]] = int(not new_value[random[0]])
    new_value[random[1]] = int(not new_value[random[1]])
    new_value[random[2]] = int(not new_value[random[2]])

    three_wrong_set[key] = new_value
```

In [8]:

```
rcParams['figure.figsize'] = 15, 5
fig, ax = plt.subplots(1, 3)

ax[0].imshow(
    np.array(one_wrong_set[1]).reshape((5,4)),
    cmap='gray_r'
)
ax[0].set_xlabel('1 bit errado', fontsize=20)

ax[1].imshow(
    np.array(two_wrong_set[1]).reshape((5,4)),
    cmap='gray_r'
)
ax[1].set_xlabel('2 bits errado', fontsize=20)
```

```
ax[2].imshow(
    np.array(three_wrong_set[1]).reshape((5,4)),
    cmap='gray_r'
)
ax[2].set_xlabel('3 bits errado', fontsize=20)

plt.show()
```



In [9]:
```
one_wrong_x, one_wrong_y = make_xy(one_wrong_set)

two_wrong_x, two_wrong_y = make_xy(two_wrong_set)

three_wrong_x, three_wrong_y = make_xy(three_wrong_set)
```

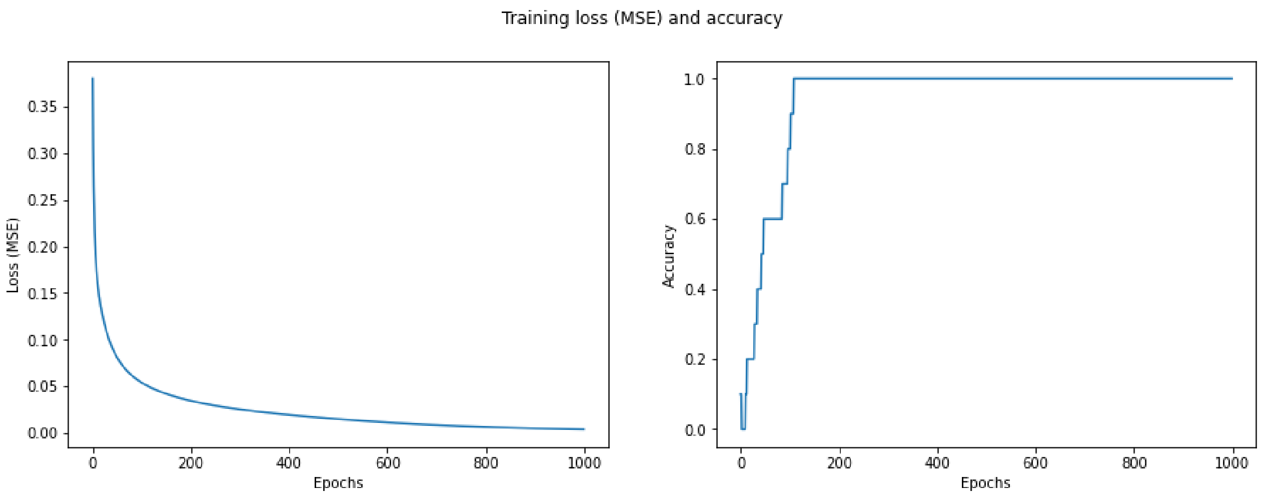
In [10]:
```
import tensorflow as tf

def build_and_evaluate_nn(hidden, activation_hidden, output, activation_output, lr, mom
    np.random.seed(10)
    tf.compat.v1.logging.set_verbosity(tf.compat.v1.logging.ERROR)

    print(f'''
    Parâmetros
    Hidden layer: {hidden}, ({activation_hidden})
    Output layer: {output}, ({activation_output})
    Learning Rate: {lr}
    Momentum: {momentum}
    ''')

    model = tf.keras.models.Sequential([
        tf.keras.layers.InputLayer(input_shape=(20, )),
        tf.keras.layers.Dense(hidden, activation=activation_hidden),
        tf.keras.layers.Dense(output, activation=activation_output)
    ])

    loss_fn = tf.keras.losses.MeanSquaredError()

    sgd = tf.keras.optimizers.SGD(
        learning_rate=lr, momentum=momentum
    )

    model.compile(
        optimizer=sgd,
        loss=loss_fn,
        metrics=["accuracy"]
    )

    history = model.fit(x, y_true, epochs=epochs, verbose=0)
    history = history.history

    rcParams['figure.figsize'] = 15, 5

    fig, (ax1, ax2) = plt.subplots(1, 2)

    fig.suptitle('Training loss (MSE) and accuracy')

    ax1.plot(history['loss'])
    ax1.set_xlabel('Epochs')
    ax1.set_ylabel('Loss (MSE)')

    ax2.plot(history['accuracy'])
    ax2.set_xlabel('Epochs')
    ax2.set_ylabel('Accuracy')

    plt.show()

    print(f'''
    Final training loss (MSE): {history['loss'][-1]}
    Final training accuracy: {history['accuracy'][-1]}
    ''')

    print(f'''
    Teste nos números com bits errados
    ''')

    print('\n\t1 bit errado:')
    model.evaluate(one_wrong_x, one_wrong_y, verbose=2)

    print('\n\t2 bit errado:')
    model.evaluate(two_wrong_x, two_wrong_y, verbose=2)

    print('\n\t3 bit errado:')
    model.evaluate(three_wrong_x, three_wrong_y, verbose=2)

    print('''
    ############################################################################
    ############################################################################
    ############################################################################
    ############################################################################
    ''')
```

In [11]:
```
parameters = [
    [0.1, 0],
    [0.4, 0],
    [0.9, 0],
    [0.1, 0.4],
    [0.9, 0.4],
]
```

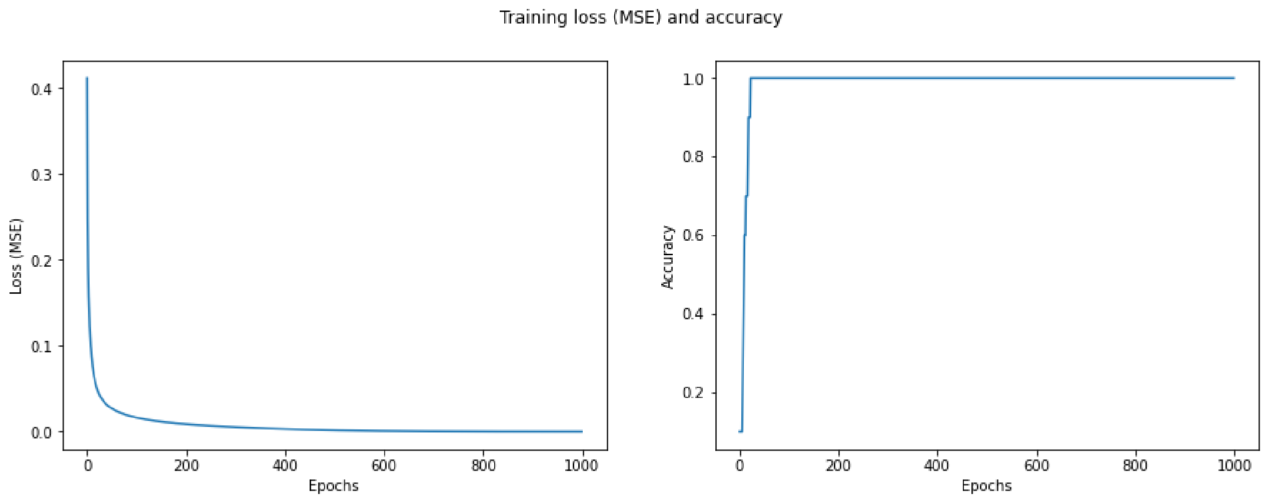
In [12]:
```
for lr, momentum in parameters:
    build_and_evaluate_nn(15, 'tanh', 10, 'linear', lr=lr, momentum=momentum)
```

```
    Parâmetros
    Hidden layer: 15, (tanh)
```

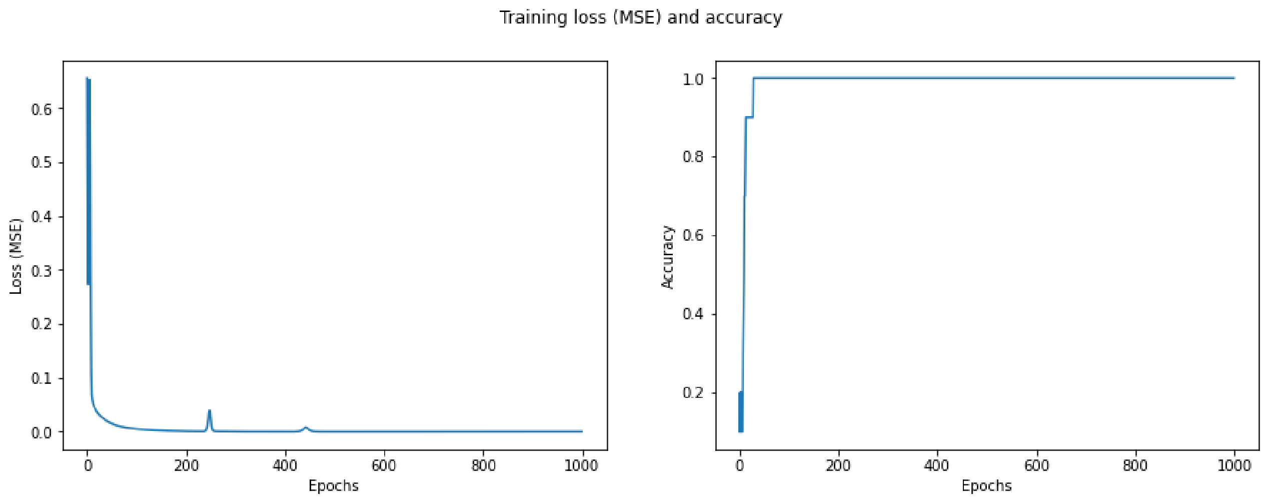
```
Output layer: 10, (linear)
Learning Rate: 0.1
Momentum: 0
```



```
    Final training loss (MSE): 0.0039929794147610664
    Final training accuracy: 1.0


    Teste nos números com bits errados


        1 bit errado:
1/1 - 0s - loss: 0.0479 - accuracy: 1.0000

        2 bit errado:
1/1 - 0s - loss: 0.0760 - accuracy: 0.8000

        3 bit errado:
1/1 - 0s - loss: 0.0949 - accuracy: 0.8000

    ############################################################################
    ############################################################################
    ############################################################################
    ############################################################################


    Parâmetros
    Hidden layer: 15, (tanh)
    Output layer: 10, (linear)
    Learning Rate: 0.4
    Momentum: 0
```

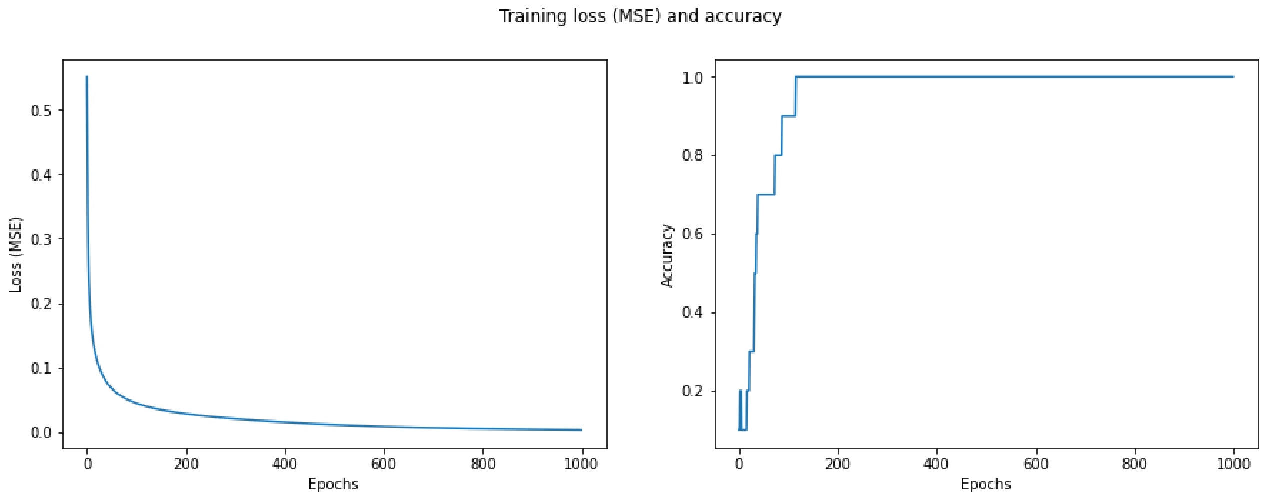


```
Final training loss (MSE): 2.227047116321046e-05
Final training accuracy: 1.0


Teste nos números com bits errados


    1 bit errado:
1/1 - 0s - loss: 0.0665 - accuracy: 0.9000

    2 bit errado:
1/1 - 0s - loss: 0.0953 - accuracy: 0.6000

    3 bit errado:
1/1 - 0s - loss: 0.1169 - accuracy: 0.7000

############################################################################
############################################################################
############################################################################
############################################################################


Parâmetros
Hidden layer: 15, (tanh)
Output layer: 10, (linear)
Learning Rate: 0.9
Momentum: 0
```



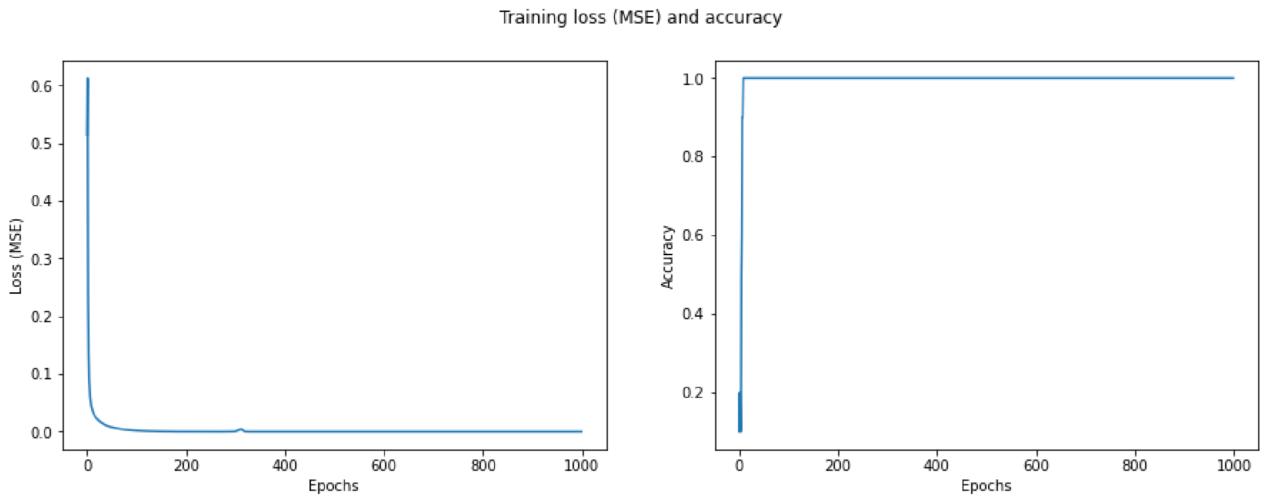
```
Final training loss (MSE): 0.00013518193736672401
Final training accuracy: 1.0
```

```
    Teste nos números com bits errados


        1 bit errado:
1/1 - 0s - loss: 0.0404 - accuracy: 0.9000

        2 bit errado:
1/1 - 0s - loss: 0.0675 - accuracy: 0.9000

        3 bit errado:
1/1 - 0s - loss: 0.0976 - accuracy: 0.7000

    ############################################################################
    ############################################################################
    ############################################################################
    ############################################################################


    Parâmetros
    Hidden layer: 15, (tanh)
    Output layer: 10, (linear)
    Learning Rate: 0.1
    Momentum: 0.4
```



```
    Final training loss (MSE): 0.002740252995863557
    Final training accuracy: 1.0


    Teste nos números com bits errados


        1 bit errado:
1/1 - 0s - loss: 0.0463 - accuracy: 1.0000

        2 bit errado:
1/1 - 0s - loss: 0.0722 - accuracy: 0.9000

        3 bit errado:
1/1 - 0s - loss: 0.0825 - accuracy: 0.7000

    ############################################################################
    ############################################################################
    ############################################################################
    ############################################################################


    Parâmetros
```

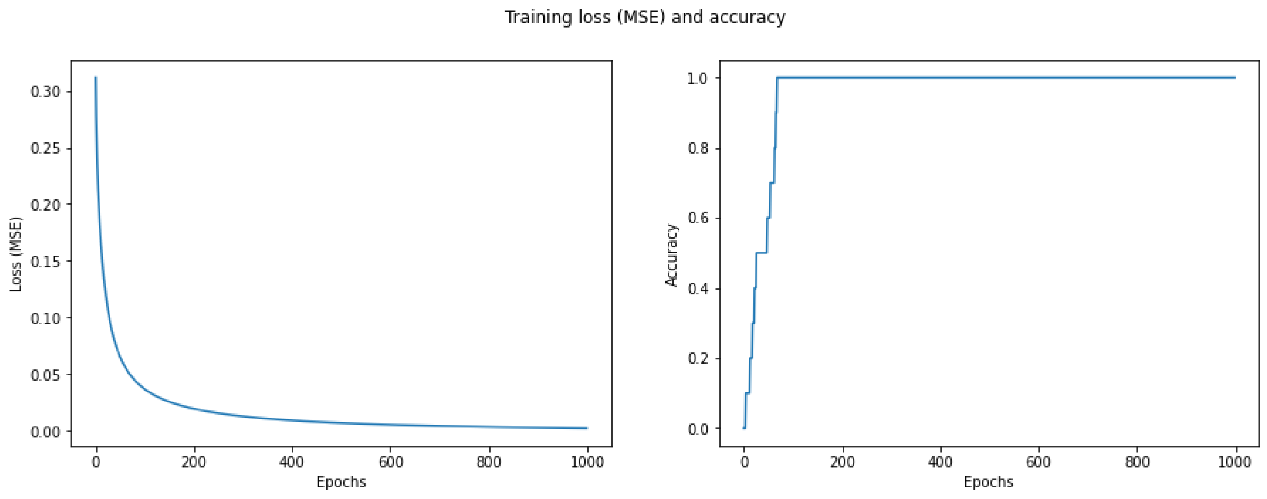
```
Hidden layer: 15, (tanh)
Output layer: 10, (linear)
Learning Rate: 0.9
Momentum: 0.4
```



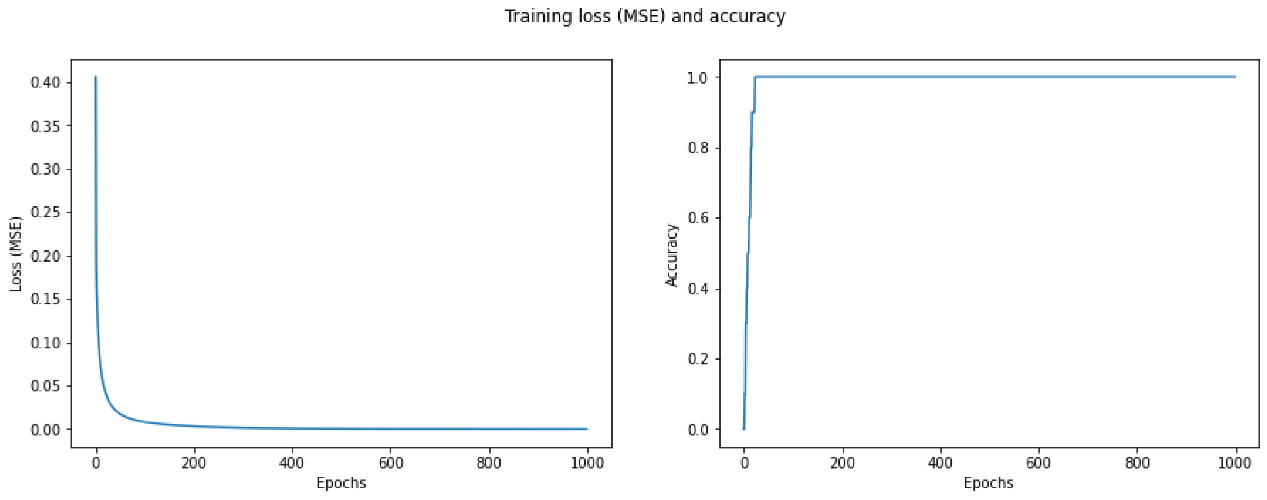
```
Final training loss (MSE): 1.205334795731719e-13
Final training accuracy: 1.0


Teste nos números com bits errados


        1 bit errado:
1/1 - 0s - loss: 0.0438 - accuracy: 0.9000

        2 bit errado:
1/1 - 0s - loss: 0.1100 - accuracy: 0.7000

        3 bit errado:
1/1 - 0s - loss: 0.1508 - accuracy: 0.7000

############################################################################
############################################################################
############################################################################
############################################################################
```

In [13]:
```python
for lr, momentum in parameters:
    build_and_evaluate_nn(25, 'tanh', 10, 'linear', lr=lr, momentum=momentum)
```

```
Parâmetros
Hidden layer: 25, (tanh)
Output layer: 10, (linear)
Learning Rate: 0.1
Momentum: 0
```

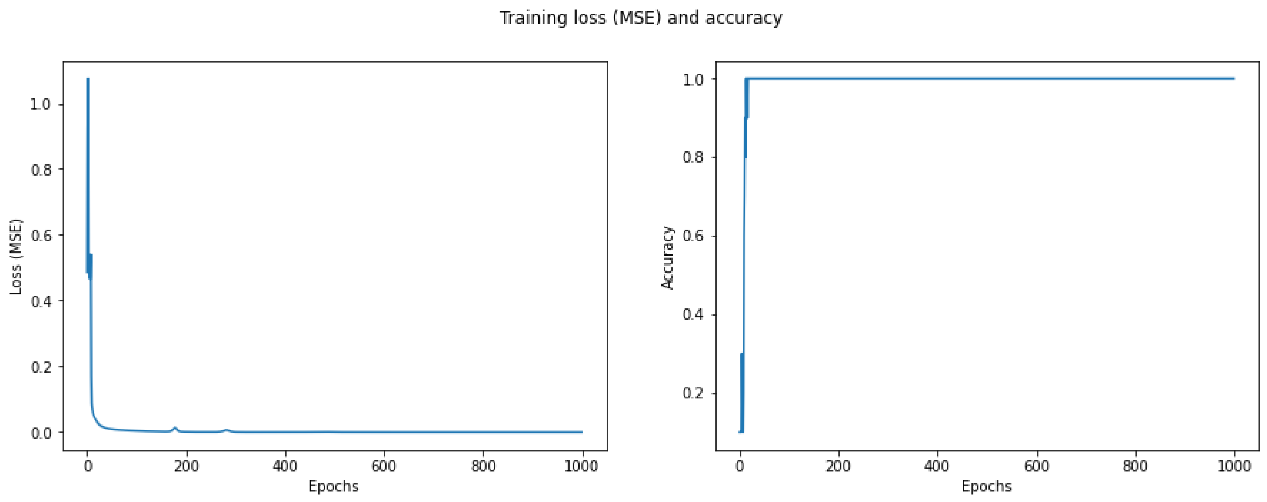


```
Final training loss (MSE): 0.0023930338211357594
Final training accuracy: 1.0


Teste nos números com bits errados


    1 bit errado:
1/1 - 0s - loss: 0.0430 - accuracy: 1.0000

    2 bit errado:
1/1 - 0s - loss: 0.0689 - accuracy: 0.9000

    3 bit errado:
1/1 - 0s - loss: 0.0914 - accuracy: 0.8000

############################################################################
############################################################################
############################################################################
############################################################################


Parâmetros
Hidden layer: 25, (tanh)
Output layer: 10, (linear)
Learning Rate: 0.4
Momentum: 0
```



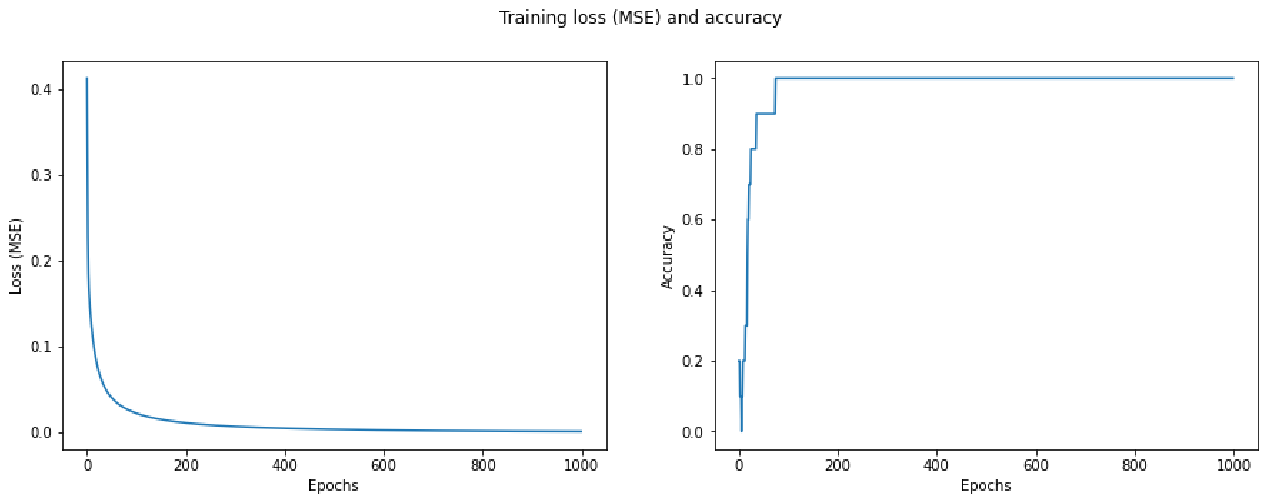
```
Final training loss (MSE): 1.476572037972801e-06
Final training accuracy: 1.0
```

```
    Teste nos números com bits errados


        1 bit errado:
1/1 - 0s - loss: 0.0773 - accuracy: 0.8000

        2 bit errado:
1/1 - 0s - loss: 0.1270 - accuracy: 0.5000

        3 bit errado:
1/1 - 0s - loss: 0.1545 - accuracy: 0.6000

    ############################################################################
    ############################################################################
    ############################################################################
    ############################################################################


    Parâmetros
    Hidden layer: 25, (tanh)
    Output layer: 10, (linear)
    Learning Rate: 0.9
    Momentum: 0
```

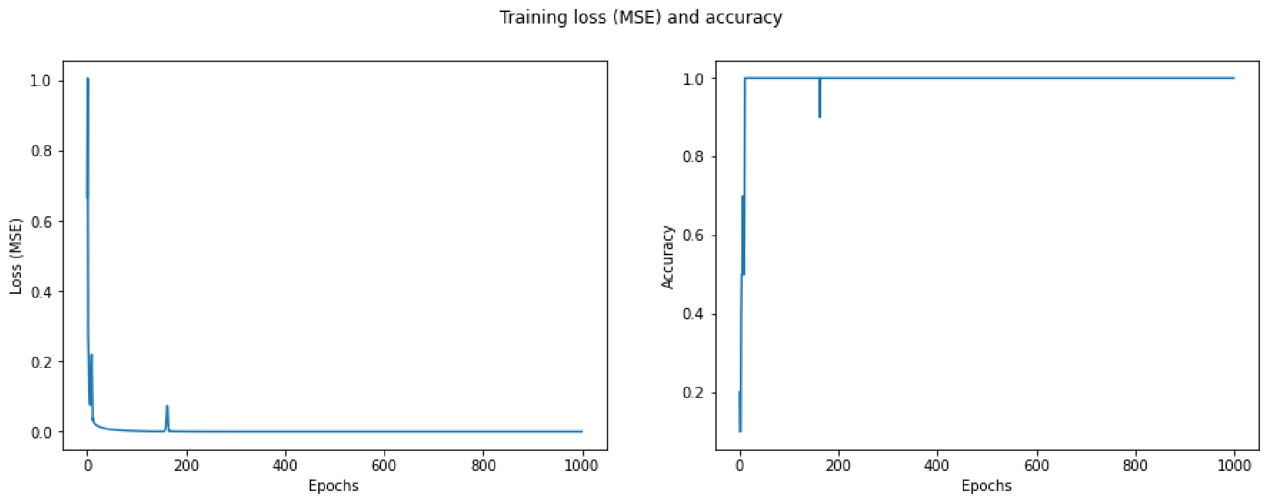


```
    Final training loss (MSE): 8.184719568760102e-10
    Final training accuracy: 1.0


    Teste nos números com bits errados


        1 bit errado:
1/1 - 0s - loss: 0.0361 - accuracy: 1.0000

        2 bit errado:
1/1 - 0s - loss: 0.0702 - accuracy: 0.9000

        3 bit errado:
1/1 - 0s - loss: 0.0728 - accuracy: 0.8000

    ############################################################################
    ############################################################################
    ############################################################################
    ############################################################################


    Parâmetros
```

```
Hidden layer: 25, (tanh)
Output layer: 10, (linear)
Learning Rate: 0.1
Momentum: 0.4
```



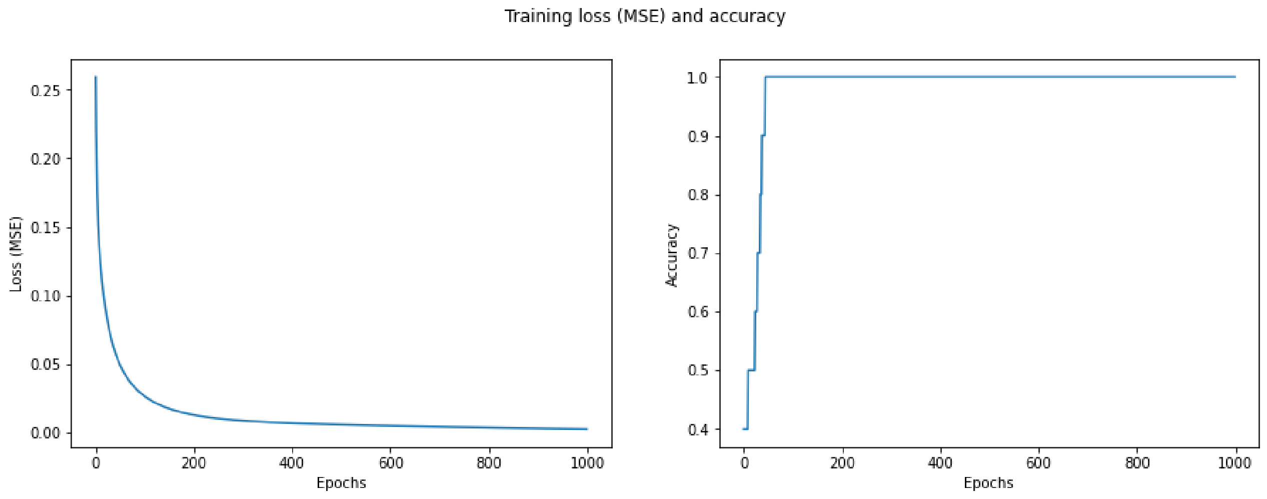
```
Final training loss (MSE): 0.00031166954431682825
Final training accuracy: 1.0


Teste nos números com bits errados


    1 bit errado:
1/1 - 0s - loss: 0.0374 - accuracy: 1.0000

    2 bit errado:
1/1 - 0s - loss: 0.0721 - accuracy: 0.8000

    3 bit errado:
1/1 - 0s - loss: 0.0723 - accuracy: 0.7000

############################################################################
############################################################################
############################################################################
############################################################################


Parâmetros
Hidden layer: 25, (tanh)
Output layer: 10, (linear)
Learning Rate: 0.9
Momentum: 0.4
```



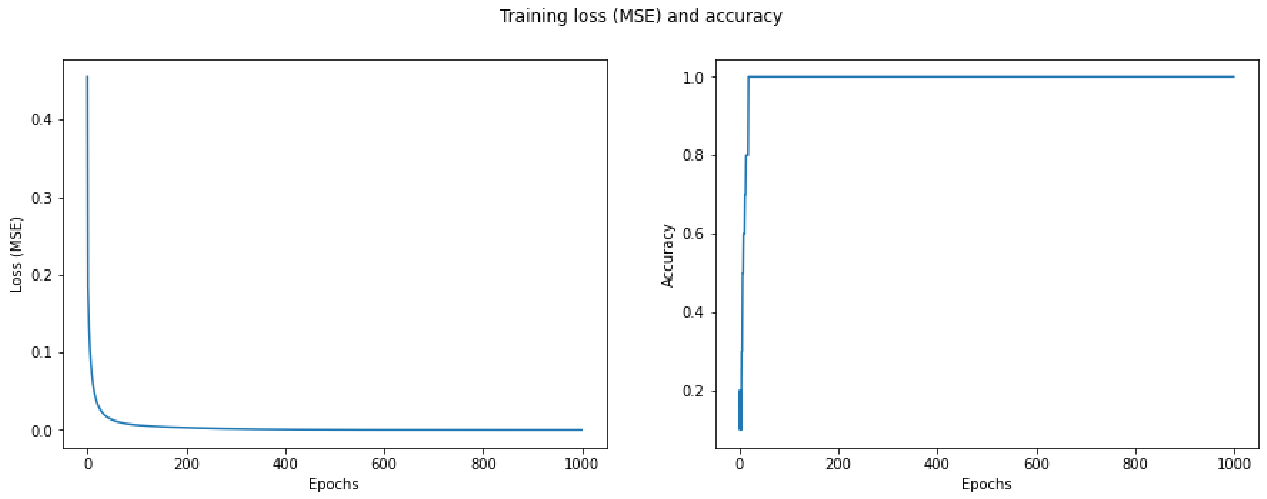
```
Final training loss (MSE): 3.12565897914608e-14
Final training accuracy: 1.0


Teste nos números com bits errados


    1 bit errado:
1/1 - 0s - loss: 0.0465 - accuracy: 0.9000

    2 bit errado:
1/1 - 0s - loss: 0.0792 - accuracy: 0.8000

    3 bit errado:
1/1 - 0s - loss: 0.1103 - accuracy: 0.7000

############################################################################
############################################################################
############################################################################
############################################################################
```

In [14]:
```python
for lr, momentum in parameters:
    build_and_evaluate_nn(35, 'tanh', 10, 'linear', lr=lr, momentum=momentum)
```

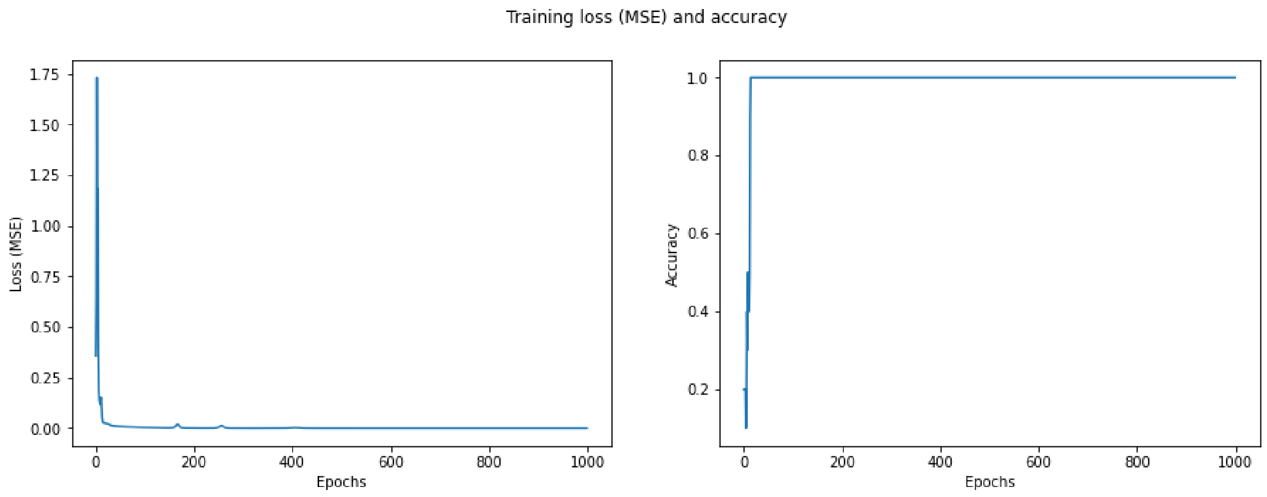
```
Parâmetros
Hidden layer: 35, (tanh)
Output layer: 10, (linear)
Learning Rate: 0.1
Momentum: 0
```



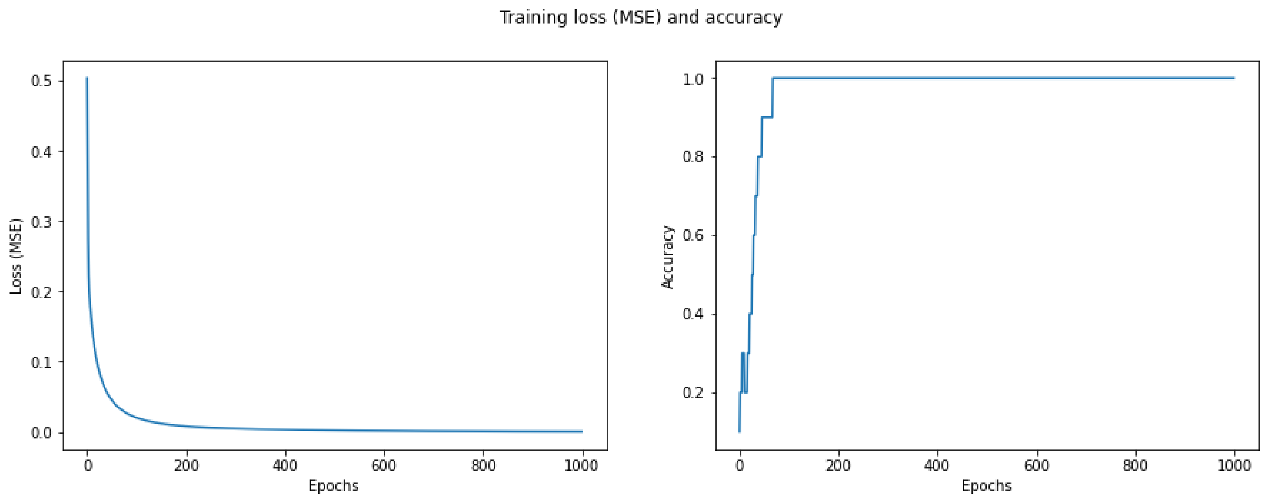
```
    Final training loss (MSE): 0.0022911611013114452
    Final training accuracy: 1.0


    Teste nos números com bits errados


        1 bit errado:
1/1 - 0s - loss: 0.0377 - accuracy: 0.9000

        2 bit errado:
1/1 - 0s - loss: 0.0850 - accuracy: 0.8000

        3 bit errado:
1/1 - 0s - loss: 0.1116 - accuracy: 0.8000

    ############################################################################
    ############################################################################
    ############################################################################
    ############################################################################


    Parâmetros
    Hidden layer: 35, (tanh)
    Output layer: 10, (linear)
    Learning Rate: 0.4
    Momentum: 0
```



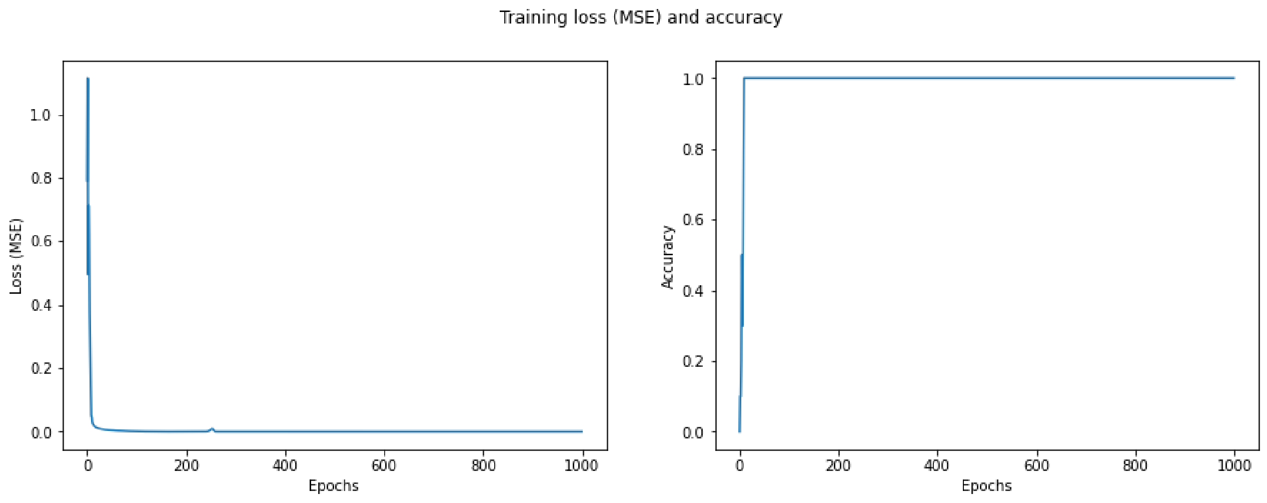
```
    Final training loss (MSE): 9.131940714723896e-07
    Final training accuracy: 1.0


    Teste nos números com bits errados


        1 bit errado:
1/1 - 0s - loss: 0.0622 - accuracy: 0.9000

        2 bit errado:
1/1 - 0s - loss: 0.1133 - accuracy: 0.8000

        3 bit errado:
1/1 - 0s - loss: 0.1489 - accuracy: 0.6000

    ############################################################################
    ############################################################################
    ############################################################################
    ############################################################################
    Parâmetros
    Hidden layer: 35, (tanh)
    Output layer: 10, (linear)
    Learning Rate: 0.9
    Momentum: 0
```



```
    Final training loss (MSE): 2.693912392714992e-06
    Final training accuracy: 1.0


    Teste nos números com bits errados


        1 bit errado:
1/1 - 0s - loss: 0.0332 - accuracy: 1.0000

        2 bit errado:
1/1 - 0s - loss: 0.0537 - accuracy: 1.0000

        3 bit errado:
1/1 - 0s - loss: 0.0695 - accuracy: 0.8000

    ############################################################################
    ############################################################################
    ############################################################################
    ############################################################################


    Parâmetros
    Hidden layer: 35, (tanh)
    Output layer: 10, (linear)
    Learning Rate: 0.1
    Momentum: 0.4
```



```
Final training loss (MSE): 0.00022103285300545394
Final training accuracy: 1.0


Teste nos números com bits errados


        1 bit errado:
1/1 - 0s - loss: 0.0418 - accuracy: 0.9000

        2 bit errado:
1/1 - 0s - loss: 0.0917 - accuracy: 0.8000

        3 bit errado:
1/1 - 0s - loss: 0.1085 - accuracy: 0.8000

    ############################################################################
    ############################################################################
    ############################################################################
    ############################################################################


    Parâmetros
    Hidden layer: 35, (tanh)
    Output layer: 10, (linear)
    Learning Rate: 0.9
    Momentum: 0.4
```



```
Final training loss (MSE): 1.889589220228742e-14
Final training accuracy: 1.0
```

```
    Teste nos números com bits errados


        1 bit errado:
1/1 - 0s - loss: 0.0451 - accuracy: 0.9000

        2 bit errado:
1/1 - 0s - loss: 0.0839 - accuracy: 0.9000

        3 bit errado:
1/1 - 0s - loss: 0.1015 - accuracy: 1.0000

    ############################################################################
    ############################################################################
    ############################################################################
    ############################################################################
```

```
In [ ]:
```