Exercise 2

In the course you learned how to do classificaiton using Fashion MNIST, a data set containing items of clothing. There's another, similar dataset called MNIST which has items of handwriting -- the digits 0 through 9.

Write an MNIST classifier that trains to 99% accuracy or above, and does it without a fixed number of epochs -- i.e. you should stop training once you reach that level of accuracy.

Some notes:

1. It should succeed in less than 10 epochs, so it is okay to change epochs= to 10, but nothing larger
2. When it reaches 99% or greater it should print out the string "Reached 99% accuracy so cancelling training!"
3. If you add any additional variables, make sure you use the same names as the ones used in the class

I've started the code for you below -- how would you finish it?

In [1]:

```
import tensorflow as tf
from os import path, getcwd, chdir

# DO NOT CHANGE THE LINE BELOW. If you are developing in a local
# environment, then grab mnist.npz from the Coursera Jupyter Notebook
# and place it inside a local folder and edit the path to that location
path = f"{getcwd()}/../tmp2/mnist.npz"
```

In [2]:

```
# GRADED FUNCTION: train_mnist
def train_mnist():
    # Please write your code only where you are indicated.
    # please do not remove # model fitting inline comments.

    class myCallback(tf.keras.callbacks.Callback):
        def on_epoch_end(self, epoch, logs={}):
            if(logs.get('acc') >= 0.99):
                print("\nReached 99% accuracy so cancelling training!")
                self.model.stop_training = True


    mnist = tf.keras.datasets.mnist

    (x_train, y_train),(x_test, y_test) = mnist.load_data(path=path)

    x_train = x_train / 255.0
    x_test = x_test / 255.0
    model = tf.keras.models.Sequential([
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(1024, activation=tf.nn.relu),
        tf.keras.layers.Dense(10, activation=tf.nn.softmax)
    ])

    model.compile(optimizer='adam',
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])

    callbacks = myCallback()
    # model fitting
    history = model.fit(x_train, y_train, epochs=10, callbacks=[callbacks]
    )
    # model fitting
    return history.epoch, history.history['acc'][-1]
```

In [3]:

```
train_mnist()
```

```
WARNING: Logging before flag parsing goes to stderr.
W0917 12:53:04.067332 139738385053504 deprecation.py:506] From /usr/local/
lib/python3.6/dist-packages/tensorflow/python/ops/init_ops.py:1251: callin
g VarianceScaling.__init__ (from tensorflow.python.ops.init_ops) with dtyp
e is deprecated and will be removed in a future version.
Instructions for updating:
Call initializer instance with the dtype argument instead of passing it to
the constructor

Epoch 1/10
60000/60000 [==============================] - 10s 162us/sample - loss: 0.
1868 - acc: 0.9437
Epoch 2/10
60000/60000 [==============================] - 9s 151us/sample - loss: 0.0
749 - acc: 0.9766
Epoch 3/10
60000/60000 [==============================] - 9s 148us/sample - loss: 0.0
484 - acc: 0.9841
Epoch 4/10
60000/60000 [==============================] - 9s 154us/sample - loss: 0.0
337 - acc: 0.9894
Epoch 5/10
59744/60000 [============================>.] - ETA: 0s - loss: 0.0271 - ac
c: 0.9911
Reached 99% accuracy so cancelling training!
60000/60000 [==============================] - 9s 148us/sample - loss: 0.0
272 - acc: 0.9911
```

Out[3]:

```
([0, 1, 2, 3, 4], 0.99105)
```

In [4]:

```
# Now click the 'Submit Assignment' button above.
# Once that is complete, please run the following two cells to save your work and close
the notebook
```

In [4]:

```
%%javascript
<!-- Save the notebook -->
IPython.notebook.save_checkpoint();
```

In [ ]:

```
%%javascript
IPython.notebook.session.delete();
window.onbeforeunload = null
setTimeout(function() { window.close(); }, 1000);
```