In this notebook, we're going to cover some of the most fundamental concepts of tensors using TensorFlow

More specifically, we're going to cover:

- Introduction to tensors
- Getting information form tensors
- Manipulating tensors
- Tensors & NumPy
- Using @tf.function(a way to speed up your refular Python funtions)
- Using GPUs with TensorFlow(or TPUs)
- Exercises to try yourself

Introduction to Tensors

```
# Import TensorFlow
import tensorflow as tf
print(tf.__version__)
```

```
2.9.2
```

```
# Create tensors with tf.constant()
scalar = tf.constant(7)
scalar
```

```
<tf.Tensor: shape=(), dtype=int32, numpy=7>
```

```
# Check the number of dimensions of a tensor(ndim stands for number of dimensions)
scalar.ndim
```

```
0
```

```
# Create a vector
vector = tf.constant([10,10])
vector
```

```
<tf.Tensor: shape=(2,), dtype=int32, numpy=array([10, 10], dtype=int32)>
```

```
# Check the dimension of our vector
vector.ndim
```

```
1
```

```
# Create a matrix (has more than 1 dimension)
matrix = tf.constant([[10, 7],
                      [7, 10]])
matrix
```

```
<tf.Tensor: shape=(2, 2), dtype=int32, numpy=
array([[10,  7],
       [ 7, 10]], dtype=int32)>
```

```
# Check the dimension of our matrix
matrix.ndim
```

```
2
```

```
# Create another matrix
another_matrix = tf.constant([[19., 7.],
                              [3., 2.],
                              [8., 9.]], dtype=tf.float16) # specify the data with
another_matrix
```

```
<tf.Tensor: shape=(3, 2), dtype=float16, numpy=
array([[19.,  7.],
       [ 3.,  2.],
       [ 8.,  9.]], dtype=float16)>
```

```
# What's the number dimensions of another_matrix?
another_matrix.ndim
```

```
2
```

```
# Let's create a tensor
tensor = tf.constant([[[1, 2, 3,],
                      [4 ,5, 6]],
                      [[7, 8, 9],
                       [10, 11, 12]],
                      [[13, 14, 15],
                     [16, 17, 18]]])
tensor
```

```
<tf.Tensor: shape=(3, 2, 3), dtype=int32, numpy=
array([[[ 1,  2,  3],
        [ 4,  5,  6]],

       [[ 7,  8,  9],
        [10, 11, 12]],

       [[13, 14, 15],
        [16, 17, 18]]], dtype=int32)>
```

```
tensor.ndim
```

```
3
```

What we've created so far:

- Scalar: a single number
- Vector: a number with direction (e.g. wind speed and direction)
- Matrix: a 2-dimensional array of numbers
- Tensor: an n-dimensional array of numbers (when n can be any number, a 0-dimensional tensor is a scalar, a 1-dimensional tensor is a vector)

Creating tensors with `tf.Variable`

```
# Create the same tensor with tf.Variable() as above
changeable_tensor = tf.Variable([10, 7])
unchangeable_tensor = tf.constant([10, 7])
changeable_tensor, unchangeable_tensor
```

```
(<tf.Variable 'Variable:0' shape=(2,) dtype=int32, numpy=array([10,  7], dtype
 <tf.Tensor: shape=(2,), dtype=int32, numpy=array([10,  7], dtype=int32)>)
```

```
# Let's try change one of the elements in our changeable tensor
changeable_tensor[0] = 7
changeable_tensor
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [42], in <cell line: 2>()
      1 # Let's try change one of the elements in our changeable tensor
----> 2 changeable_tensor[0] = 7
      3 changeable_tensor

TypeError: 'ResourceVariable' object does not support item assignment
```

SEARCH STACK OVERFLOW

```
# How about we try .assign()
changeable_tensor[0].assign(7)
changeable_tensor
```

```
<tf.Variable 'Variable:0' shape=(2,) dtype=int32, numpy=array([7, 7], dtype=in
```

```
# let's try change our unchangeable tensor
unchangeable_tensor[0].assign(7)
unchangeable_tensor
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Input In [55], in <cell line: 2>()
      1 # let's try change our unchangeable tensor
----> 2 unchangeable_tensor[0].assign(7)
      3 unchangeable_tensor

File ~/miniforge3/envs/tensorflow/lib/python3.8/site-packages/tensorflow/pythc
    437 if name in {"T", "astype", "ravel", "transpose", "reshape", "clip", "s
    438             "tolist", "data"}:
    439   # TODO(wangpeng): Export the enable_numpy_behavior knob
    440   raise AttributeError(
    441       f"{type(self).__name__} object has no attribute '{name}'. " + ""
    442     If you are looking for numpy-related methods, please run the follc
    443     from tensorflow.python.ops.numpy_ops import np_config
    444     np_config.enable_numpy_behavior()
    445   """)
--> 446 self.__getattribute__(name)
```

🔑 **Notes:**Rarely in practive will you need to decide whether to use `tf.constant` or `tf.Variable` to create tensors, as TensorFlow does this for you. However, if in doubt, use `tf.constant` and change it later if needed.