```
import tensorflow as tf
print("tensorflow version is :",tf.__version__)
```

```
tensorflow version is : 2.13.0
```

```
scaler = tf.constant(24)
scaler
```

```
<tf.Tensor: shape=(), dtype=int32, numpy=24>
```

```
vector = tf.constant([1,2,3])
vector
```

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

```
vector1 = tf.constant([1.1,2,3.1111])
vector1
```

```
<tf.Tensor: shape=(3,), dtype=float32, numpy=array([1.1   , 2.    , 3.1111], dtype=float32)>
```

```
vector1.ndim
```

```
1
```

```
metrix = tf.constant([[1,2,3],[3,4,5],[5,6,7]]);
metrix
```

```
<tf.Tensor: shape=(3, 3), dtype=int32, numpy=
array([[1, 2, 3],
       [3, 4, 5],
       [5, 6, 7]], dtype=int32)>
```

```
metrix.ndim
```

```
2
```

```
metrix2 = tf.constant([[1.,2.,3.],[2.,3.,4.,]] ,dtype=tf.float16)
metrix2
```

```
<tf.Tensor: shape=(2, 3), dtype=float16, numpy=
array([[1., 2., 3.],
       [2., 3., 4.]], dtype=float16)>
```

```
metrix2.ndim
```

```
2
```

```
metrix3 = tf.constant([[2,3,4],[3,4,5],[5,6,7]], dtype=tf.float16)
metrix3
```

```
<tf.Tensor: shape=(3, 3), dtype=float16, numpy=
array([[2., 3., 4.],
       [3., 4., 5.],
       [5., 6., 7.]], dtype=float16)>
```

```
metrix4 = tf.constant([[1.,2.,3.],[3.,4.,5.],[5.,6.,7.]])
metrix4
```

```
<tf.Tensor: shape=(3, 3), dtype=float32, numpy=
array([[1., 2., 3.],
       [3., 4., 5.],
       [5., 6., 7.]], dtype=float32)>
```

```
tensor = tf.constant([
    [[1,2,3],[2,3,4]],
    [[4,5,6],[5,6,7]],
    [[6,7,8],[7,8,9]]
  ])
tensor
```

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

       [[4, 5, 6],
        [5, 6, 7]],

       [[6, 7, 8],
        [7, 8, 9]]], dtype=int32)>
```

```
tensor.ndim
```

```
3
```

Variable Tutorial Here

```
var_vector = tf.Variable([1,3,5])
var_vector
```

```
<tf.Variable 'Variable:0' shape=(3,) dtype=int32, numpy=array([1, 3, 5], dtype=int32)>
```

```
const_vector = tf.constant([1,3,5])
const_vector
```

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

```
var_vector
```

```
<tf.Variable 'Variable:0' shape=(3,) dtype=int32, numpy=array([1, 3, 5], dtype=int32)>
```

```
var_vector[1].assign(9)
var_vector
```

```
<tf.Variable 'Variable:0' shape=(3,) dtype=int32, numpy=array([1, 9, 5], dtype=int32)>
```

```
const_vector[1].assign(9)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-27-da39ca31b9f8> in <cell line: 1>()
----> 1 const_vector[1].assign(9)

/usr/local/lib/python3.10/dist-packages/tensorflow/python/framework/ops.py in __getattr__(self, name)
    428         np_config.enable_numpy_behavior()
    429       """)
--> 430     self.__getattribute__(name)
    431
    432   @property

AttributeError: 'tensorflow.python.framework.ops.EagerTensor' object has no attribute 'assign'

SEARCH STACK OVERFLOW
```

```
var_metrix = tf.Variable([[1,2,3],[3,4,5]])
var_metrix
```

```
<tf.Variable 'Variable:0' shape=(2, 3) dtype=int32, numpy=
array([[1, 2, 3],
       [3, 4, 5]], dtype=int32)>
```

Converting python list and arrays into tensor

```
import numpy as np
```

```
scaler = 10
vector = [1,2,3,4]
array = np.array([10,20,30,40])
matrix = np.array([[10,20,30],[30,40,50]])


print("array :",array)
print("matrix :",matrix)
print("scaler :",scaler)
print("vector :",vector)
```

```
array : [10 20 30 40]
matrix : [[10 20 30]
 [30 40 50]]
scaler : 10
vector : [1, 2, 3, 4]
```

```
tfscaler = tf.convert_to_tensor(scaler)
tfscaler
```

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

```
tfvector = tf.convert_to_tensor(vector)
tfvector
```

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

```
tfarray = tf.convert_to_tensor(array)
tfarray
```

```
<tf.Tensor: shape=(4,), dtype=int64, numpy=array([10, 20, 30, 40])>
```

```
tfmatrix = tf.convert_to_tensor(matrix)
tfmatrix
```

```
<tf.Tensor: shape=(2, 3), dtype=int64, numpy=
array([[10, 20, 30],
       [30, 40, 50]])>
```

Creating RANDOM tensor

```
tensor = tf.random.normal(shape=(3,2),mean=10.0,stddev=2.0,dtype=tf.float16)
tensor
```

```
<tf.Tensor: shape=(3, 2), dtype=float16, numpy=
array([[13.84 , 10.51 ],
       [ 7.273,  8.63 ],
       [11.62 , 12.68 ]], dtype=float16)>
```

```
tensor = tf.random.uniform(shape=(3,2),minval=1.0,maxval=2.0,dtype=tf.float16)
tensor
```

```
<tf.Tensor: shape=(3, 2), dtype=float16, numpy=
array([[1.752, 1.568],
       [1.595, 1.133],
       [1.063, 1.185]], dtype=float16)>
```

Getting Basic Information from tensors

```
tensor = tf.constant([[[1,2,3],[2,3,4]],
                      [[4,5,6],[5,6,7]],
                      [[6,7,8],[7,8,9]]
                      ])
tensor
```

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

       [[4, 5, 6],
        [5, 6, 7]],

       [[6, 7, 8],
        [7, 8, 9]]], dtype=int32)>
```

```
# getting data type of any tensor
print("data type of the tensor :",tensor.dtype)
```

```
data type of the tensor : <dtype: 'int32'>
```

```
# getting shape of any tensor
print("shape of the tensor :", tensor.shape)
```

```
shape of the tensor : (3, 2, 3)
```

```
# rank or number of dimension of a tensor
print("dimension of a tensor :",tensor.ndim)
```

```
dimension of a tensor : 3
```

```
# total number of element in a tensor
print("Total number of element in a tensor :",tf.size(tensor))
```

```
Total number of element in a tensor : tf.Tensor(18, shape=(), dtype=int32)
```

```
# fetch the value from tensor
print("Total number of element in a tensor :",tf.size(tensor).numpy())
```

```
Total number of element in a tensor : 18
```

```
tensor.numpy()
```

```
array([[[1, 2, 3],
        [2, 3, 4]],

       [[4, 5, 6],
        [5, 6, 7]],

       [[6, 7, 8],
        [7, 8, 9]]], dtype=int32)
```