Goal:

PyTorch is one of the most popular library for deep learning project. We'll explore PyTorch in detail PyTorch and how it works

Scalar has zero dimensions.It is a single number. Vector is two dimensional. Matrix is two or more scalar.So anything more than one dimension can be called as tensor.

tensors are similar to numpy array.However numpy array can't be used with GPU.This is the main a array.Now let's see some numpy array and tensors:

```
import numpy as np
import torch


np_arr = np.array(3)
tensor = torch.tensor(5)
print(np_arr)
print(tensor)
print(type(np_arr))
print(type(tensor))
```

```
3
tensor(5)
<class 'numpy.ndarray'>
<class 'torch.Tensor'>
```

We can convert numpy array to tensor and viceversa.

```
tensor_from_np = torch.from_numpy(np_arr)
print(tensor_from_np)
print(type(tensor_from_np))
```

```
tensor(3)
<class 'torch.Tensor'>
```

```
np_arr_from_tensor = tensor.numpy()
print(np_arr_from_tensor)
print(type(np_arr_from_tensor))
```

```
5
<class 'numpy.ndarray'>
```

```
np_arr = np.array(3)
tensor = torch.tensor(5)
```

Matrix Initialization and Matrix Operations

using Numpy:

```
np.random.seed(0)
mat1 = np.random.randn(2,2)
mat2 = np.array([[1,2],[3,4]])
print(mat1)
print(type(mat1))
print(mat2)
print(type(mat2))
```

```
[[1.76405235 0.40015721]
 [0.97873798 2.2408932 ]]
<class 'numpy.ndarray'>
[[1 2]
 [3 4]]
<class 'numpy.ndarray'>
```

```
print(mat1+mat2)
print(mat1-mat2)
print(mat1/mat2)
print(mat1*mat2)
```

```
[[2.76405235 2.40015721]
 [3.97873798 6.2408932 ]]
[[ 0.76405235 -1.59984279]
 [-2.02126202 -1.7591068 ]]
[[1.76405235 0.2000786 ]
 [0.32624599 0.5602233 ]]
[[1.76405235 0.80031442]
 [2.93621395 8.9635728 ]]
```

using torch tensor:

```
torch.random.seed = 0
mat1 = torch.randn(2,2)
mat2 = torch.from_numpy(np.array([[1,2],[3,4]]))
print(mat1)
print(type(mat1))
print(mat2)
print(type(mat2))
print(mat2.shape)
```

```
tensor([[-1.1677,  1.4882],
        [ 0.1569, -1.6925]])
<class 'torch.Tensor'>
tensor([[1, 2],
        [3, 4]])
<class 'torch.Tensor'>
torch.Size([2, 2])
```

```
print(mat1+mat2)
print(mat1-mat2)
print(mat1/mat2)
print(mat1*mat2)
```

```
tensor([[-0.1677,  3.4882],
        [ 3.1569,  2.3075]])
tensor([[-2.1677, -0.5118],
        [-2.8431, -5.6925]])
tensor([[-1.1677,  0.7441],
        [ 0.0523, -0.4231]])
tensor([[-1.1677,  2.9763],
        [ 0.4708, -6.7699]])
```

As we can see numpy array and torch tensors are very similar in usage.