Manipulate PyTorch Tensors

Matrix manipulation

In [1]:

```
import torch
```

Make the matrices A and B below. Add them together to obtain a matrix C. Print these three matrices.

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \qquad B = \begin{bmatrix} 10 & 20 \\ 30 & 40 \end{bmatrix} \qquad C = A + B = ?$$

In [2]:

```
# write your code here

A = torch.tensor([[1., 2.], [3., 4.]])
B = torch.tensor([[10., 20.], [30., 40.]])
C = A + B

# print
print(A)
print('')
print(B)
print('')
print(C)
```

```
tensor([[1., 2.],
        [3., 4.]])

tensor([[10., 20.],
        [30., 40.]])

tensor([[11., 22.],
        [33., 44.]])
```

Print the dimension, size and type of the matrix A. Remember, the commands are dim(), size() and type()

In [3]:

```
# write your code here

print(A.dim())    # print the dimension of the matrix A
print('')
print(A.size())    # print the size of the matrix A
print('')
print(A.type())    # print the type of the matrix A
```

```
2

torch.Size([2, 2])

torch.FloatTensor
```

Convert the matrix A to be an integer matrix (type LongTensor). Remember, the command is long(). Then print the type to check it was indeed converted.

In [4]:

```
# write your code here

A_long = A.long()

print(A.type())    # print the type of A_long
print('')
print(A_long.type())    # print the type of A
```

```
torch.FloatTensor

torch.LongTensor
```

Make a random 5 x 2 x 3 Tensor. The command is torch.rand. Then do the following: 1) Print the tensor, 2) Print its type, 3) Print its dimension, 4) Print its size, 5) Print the size of its middle dimension.

In [5]:

```
# write your code here

A = torch.rand(5,2,3)

print(A)
print(A.type())    # print the type of A
print(A.dim())    # print the dimension of A
print(A.size())    # print the size of A
print(A[0].size())    # print the size of the middle (second) dimension
```

```
tensor([[[0.5338, 0.7103, 0.8205],
         [0.5245, 0.6875, 0.6619]],

        [[0.2573, 0.7673, 0.8258],
         [0.3326, 0.7691, 0.8524]],

        [[0.1188, 0.0775, 0.8690],
         [0.2040, 0.3212, 0.5377]],

        [[0.9709, 0.0436, 0.6935],
         [0.7388, 0.3224, 0.9653]],

        [[0.7667, 0.3031, 0.0613],
         [0.9679, 0.4045, 0.6809]]])
torch.FloatTensor
3
torch.Size([5, 2, 3])
torch.Size([2, 3])
```

Make 2 x 3 x 4 x 5 tensor filled with zeros then print it. (The command is torch.zeros). See if you can make sense of the display.

In [6]:

```
# write your code here

A = torch.zeros(2, 3, 4, 5)

print(A)
```

```
tensor([[[[0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0.]],

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

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


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

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

         [[0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0.]]]])
```