Manipulate PyTorch Tensors

Matrix manipulation

In [14]:
```
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 [15]:
```
# write your code here

A = torch.tensor([
                  [1,2],
                  [3,4]
], dtype=torch.int32)
B = torch.tensor([
                  [10,20],
                  [30,40]
], dtype=torch.int32)
C = torch.add(A, B)


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

```
tensor([[1, 2],
        [3, 4]], dtype=torch.int32)

tensor([[10, 20],
        [30, 40]], dtype=torch.int32)

tensor([[11, 22],
        [33, 44]], dtype=torch.int32)
```

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

In [16]:
```

# 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.IntTensor
```

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 [17]:
```

# write your code here

A_long = A.long()

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

```
torch.LongTensor

torch.IntTensor
```

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 [18]:
```

# 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.size()[1])    # print the size of the middle (second) dimension
```

```
tensor([[[0.3645, 0.1433, 0.8571],
         [0.0463, 0.3748, 0.7066]],

        [[0.1150, 0.8077, 0.4308],
         [0.8483, 0.6448, 0.4286]],

        [[0.8980, 0.8588, 0.7806],
         [0.8119, 0.3389, 0.9791]],

        [[0.7925, 0.7047, 0.8693],
         [0.3110, 0.7259, 0.1317]],

        [[0.2784, 0.7464, 0.5447],
         [0.3334, 0.2394, 0.5406]]])
torch.FloatTensor
3
torch.Size([5, 2, 3])
2
```

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 [19]:
```
# 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.]]]])
```

In [19]: