Manipulate PyTorch Tensors

Matrix manipulation

```
import torch
```

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

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

```
# 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()

```

# 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.

```
1
2   # write your code here
3
4   A_long = A.long()
5
6   print(A_long.type())    # print the type of A_long
7   print('')
8   print(A.type())    # print the type of A
```

```
torch.LongTensor

torch.FloatTensor
```

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.

```
1
2   # write your code here
3
4   A = torch.rand(5,2,3)
5
6   print(A)
7   print(A.type())    # print the type of A
8   print(A.dim())    # print the dimension of A
9   print(A.size())    # print the size of A
10  print(A.size(dim=2))    # print the size of the middle (second) dimension
```

```
tensor([[[0.9017, 0.5611, 0.3273],
         [0.0435, 0.3761, 0.0293]],

        [[0.8545, 0.3448, 0.6490],
         [0.0759, 0.6706, 0.0460]],

        [[0.5566, 0.2830, 0.6936],
         [0.2322, 0.9143, 0.1310]],

        [[0.5235, 0.3920, 0.2956],
         [0.3812, 0.3474, 0.0411]],

        [[0.1972, 0.8266, 0.3053],
         [0.8443, 0.9433, 0.3086]]])
torch.FloatTensor
3
torch.Size([5, 2, 3])
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.

```
1
2   # write your code here
```

```
3
4  A = torch.zeros(2,3,4,5)
5
6  print(A)
7
```

```
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.]]]])
```