In [2]:
```
import torch
```
In [3]:
```
x = torch.tensor(3.0)
y = torch.tensor(2.0)

x+y, x*y, x/y, x**y
```
Out[3]:
```
(tensor(5.), tensor(6.), tensor(1.5000), tensor(9.))
```
In [4]:
```
x = torch.arange(3)
x
```
Out[4]:
```
tensor([0, 1, 2])
```
In [5]:
```
x[2]
```
Out[5]:
```
tensor(2)
```
In [6]:
```
len(x)
```
Out[6]:
```
3
```
In [7]:
```
x.shape
```
Out[7]:
```
torch.Size([3])
```
In [8]:
```
A = torch.arange(6).reshape(3,2)
A
```
Out[8]:
```
tensor([[0, 1],
        [2, 3],
        [4, 5]])
```
In [9]:
```
A.T
```
Out[9]:
```
tensor([[0, 2, 4],
        [1, 3, 5]])
```
In [10]:
```
A = torch.tensor([[1, 2, 3], [2, 0, 4], [3, 4, 5]])
A == A.T
```
Out[10]:
```
tensor([[True, True, True],
        [True, True, True],
        [True, True, True]])
```
In [11]:
```
torch.arange(24).reshape(2,3,4)
```
Out[11]:
```
tensor([[[ 0,  1,  2,  3],
         [ 4,  5,  6,  7],
         [ 8,  9, 10, 11]],

        [[12, 13, 14, 15],
         [16, 17, 18, 19],
         [20, 21, 22, 23]]])
```
In [12]:
```
A = torch.arange(6, dtype=torch.float32).reshape(2,3)
B = A.clone()
A, A+B
```
Out[12]:
```
(tensor([[0., 1., 2.],
         [3., 4., 5.]]),
 tensor([[ 0.,  2.,  4.],
         [ 6.,  8., 10.]]))
```
In [13]:
```
A*B
```
Out[13]:
```
tensor([[ 0.,  1.,  4.],
        [ 9., 16., 25.]])
```
In [14]:
```
a = 2
X = torch.arange(24).reshape(2,3,4)
a+X, (a*x).shape
```

Out[14]:
```
(tensor([[[ 2,  3,  4,  5],
          [ 6,  7,  8,  9],
          [10, 11, 12, 13]],

         [[14, 15, 16, 17],
          [18, 19, 20, 21],
          [22, 23, 24, 25]]]),
 torch.Size([3]))
```

In [15]:
```
x = torch.arange(3, dtype=torch.float32)
x, x.sum()
```

Out[15]:
```
(tensor([0., 1., 2.]), tensor(3.))
```

In [16]:
```
A.shape, A.sum()
```

Out[16]:
```
(torch.Size([2, 3]), tensor(15.))
```

In [17]:
```
A.shape, A.sum(axis=0).shape
```

Out[17]:
```
(torch.Size([2, 3]), torch.Size([3]))
```

In [18]:
```
A.shape, A.sum(axis=1).shape
```

Out[18]:
```
(torch.Size([2, 3]), torch.Size([2]))
```

In [20]:
```
A.sum(axis=[0,1]) == A.sum()
```

Out[20]:
```
tensor(True)
```

In [21]:
```
A.mean(), A.sum() / A.numel()
```

Out[21]:
```
(tensor(2.5000), tensor(2.5000))
```

In [22]:
```
A.mean(axis=0), A.sum(axis=0) / A.shape[0]
```

Out[22]:
```
(tensor([1.5000, 2.5000, 3.5000]), tensor([1.5000, 2.5000, 3.5000]))
```

In [23]:
```
sum_A = A.sum(axis=1, keepdims=True)
sum_A, sum_A.shape
```

Out[23]:
```
(tensor([[ 3.],
         [12.]]),
 torch.Size([2, 1]))
```

In [24]:
```
A/sum_A
```

Out[24]:
```
tensor([[0.0000, 0.3333, 0.6667],
        [0.2500, 0.3333, 0.4167]])
```

In [25]:
```
A.cumsum(axis=0)
```

Out[25]:
```
tensor([[0., 1., 2.],
        [3., 5., 7.]])
```

In [26]:
```
y = torch.ones(3, dtype=torch.float32)
x,y,torch.dot(x,y)
```

Out[26]:
```
(tensor([0., 1., 2.]), tensor([1., 1., 1.]), tensor(3.))
```

In [27]:
```
torch.sum(x*y)
```

Out[27]:
```
tensor(3.)
```

In [29]:
```
A.shape, x.shape, torch.mv(A,x), A@x
```

Out[29]:
```
(torch.Size([2, 3]), torch.Size([3]), tensor([ 5., 14.]), tensor([ 5., 14.]))
```

In [30]:
```
B = torch.ones(3,4)
torch.mm(A,B), A@B
```

Out[30]:
```
(tensor([[ 3.,  3.,  3.,  3.],
         [12., 12., 12., 12.]]),
 tensor([[ 3.,  3.,  3.,  3.],
         [12., 12., 12., 12.]]))
```

In [31]:
```
u = torch.tensor([3.0,-4.0])
torch.norm(u)
```

Out[31]:
```
tensor(5.)
```

In [32]:
```
torch.abs(u).sum()
```

Out[32]:
```
tensor(7.)
```

In [34]:
```
torch.norm(torch.ones((4,9)))
```

Out[34]:
```
tensor(6.)
```

Discussion: Linear algebra was one of the first subjects I studied in my first semester, but unfortunately, I wasn't very focused on my studies back then, so there are many gaps in my understanding of the concepts. Nevertheless, while working on this unit's exercises, I was able to revisit parts that I had forgotten, such as matrix transposition. The most useful part of this unit was realizing how easily matrix-vector and matrix-matrix multiplication can be done using the "@" operator.