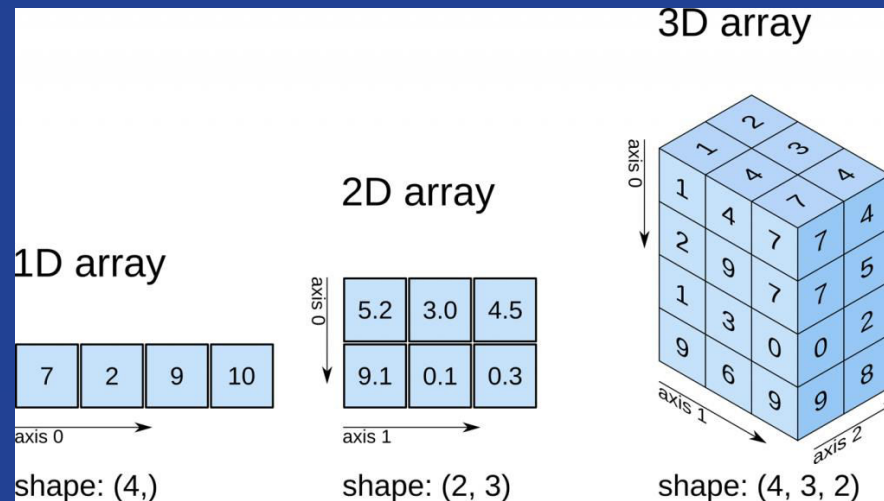
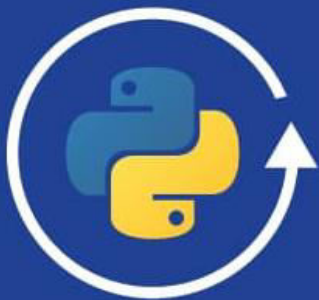


# “Tự Học Lập Trình Python “

## Bài 38: Numpy Array

### Xử lý mảng đa chiều python với numpy



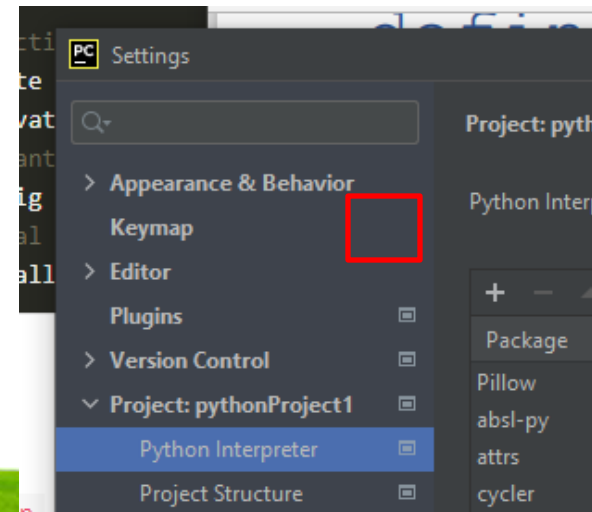
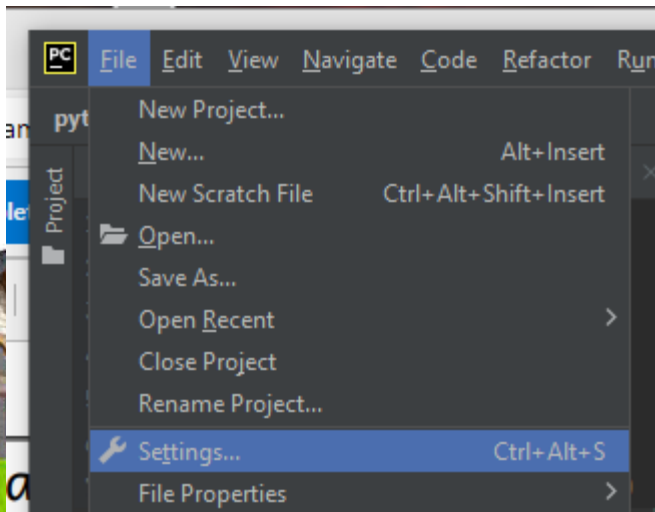
## 1. *Khái niệm :*

- \* NumPy là viết tắt của Numerical Python
- \* NumPy là một thư viện Python được sử dụng để làm việc với các mảng, ma trận.
- \* NumPy nhằm mục đích cung cấp một đối tượng mảng nhanh hơn tới 50 lần so với danh sách Python truyền thống.

## 2. *Cài đặt thư viện numpy trên pycharm :*

Cách 1: `pip install numpy`

Cách 2:



3. Sử dụng *import* để gọi thư viện : *import numpy as np*

4 . Sử dụng *np.array* để convert 1 list hoặc tuple sang array

# From List:

```
L = [1, 9, 4]
a = np.array(L)
print(a)
print(type(a))
>>> [1 9 4]
>>> <class 'numpy.ndarray'>
```

# From Tuple:

```
L = (1, 9, 4)
a = np.array(L)
print(a)
print(type(a))
>>> [1 9 4]
>>> <class 'numpy.ndarray'>
```



## 5. Thử so sánh list thông thường và array trong numpy

```
import numpy as np
arr = [
    [16, 4, 7],
    [15, 8, 17],
    [17, 1, 16],
    [17, 7, 19]
]
```

```
print(arr) # xuất list thông thường
a = np.array(arr)
print(a) # xuất array numpy
```

```
[[16, 4, 7], [15, 8, 17], [17, 1, 16], [17, 7, 19]]
[[16  4  7]
 [15  8 17]
 [17  1 16]
 [17  7 19]]
```



## 6. Các hàm cơ bản: *ndim* , *shape* , *size*

```
import numpy as np
```

```
arr = [  
    [16, 4, 7],  
    [15, 8, 17],  
    [17, 1, 16],  
    [17, 7, 19]  
]
```

```
a=np.array(arr)
```

```
# kiểm tra mảng 1 chiều, 2 chiều ...
```

```
print("Số chiều của mảng : ",a.ndim)
```

```
# kiểm tra mảng có bao nhiêu (dòng, cột)
```

```
print("Size: ", a.shape)
```

```
# kiểm tra số phần tử của mảng
```

```
print("Amount of element: ", a.size)
```





## 7. Tạo array từ 1 list có sẵn : *reshape*

```
import numpy as np
L = [1, 2, 3, 4, 5, 6, 7, 8]
# định dạng lại list theo arr(dòng, cột)
x = np.array(L).reshape(2, 4)
print(x)
```

```
[[1 2 3 4]
 [5 6 7 8]]
```

Chú ý : Sẽ lỗi nếu shape không tương ứng với số phần tử cấu thành .

```
import numpy as np
L = [1, 2, 3, 4, 5, 6, 7, 8]
# định dạng lại list theo arr(dòng, cột)
x = np.array(L).reshape(2, 3)
print(x)
```

```
x = np.array(L).reshape(2, 3)
ValueError: cannot reshape array of size 8 into shape (2,3)
```



### 8. Tạo array từ 1 dãy theo (begin,end,step)

```
z = np.arange(0, 100, 2)
print(z)
```

```
[ 0  2  4  6  8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46
 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94
 96 98]
```

### 9 Tạo mảng mà các phần tử cách đều nhau

*Arr phần tử từ 1 đến 10, số phần tử bằng 5*

```
z=np.linspace(1, 10,5)
print(z)
```

```
[ 1.    3.25  5.5   7.75 10. ]
```

*Arr các phần tử từ 1 đến 10, số phần tử bằng 10*

```
z=np.linspace(1, 10,10)
print(z)
```

```
[ 1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
```



## 9. Tạo array từ gồm toàn số 0 hoặc 1

`z = np.zeros((3,5))` # (3 hàng, 5 cột)

```
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
```

`z = np.ones((2,4))` # (2 hàng 4 cột)  
`print(z)`

```
[[1. 1. 1. 1.]
 [1. 1. 1. 1.]]
```





## 10. Access phần tử của array

```
arr = [
    [16, 4, 7],
    [15, 8, 17],
    [17, 1, 16],
    [17, 7, 19]
]
a = np.array(arr)
```

#1 access 1 phần tử tại dòng, cột :  
a[ i, j ] hoặc a[ i ][ j ]

`print(a[0,1])` # cách 1: dòng 0, cột 1  
`print(a[0][1])` # cách 2

```
arr = [
    [16, 4, 7],
    [15, 8, 17],
    [17, 1, 16],
    [17, 7, 19]
]
```

#2 access nhiều phần tử theo dòng và cột

`print(a[0,:])` # xuất dòng index 0  
`print(a[:, -1])` # xuất cột cuối cùng

	0	1	2
0	16	4	7
1	15	8	17
2	17	1	16
3	17	7	19

#3 xuất phần tử thuộc dòng 1 đến 2, nằm trong cột 3

`print(a[1:3,1])`

	0	1	2
0	16	4	7
1	15	8	17
2	17	1	16
3	17	7	19

## 11. Replace phần tử của array

```
arr = [
    [16, 4, 7],
    [15, 8, 17],
    [17, 1, 16],
    [17, 7, 19]
]
a = np.array(arr)
```

#1 replace 1 phần tử tại dòng, cột :

```
a[0,1] = 100
print(a)
```

```
arr = [
    [16, 4, 7],
    [15, 8, 17],
    [17, 1, 16],
    [17, 7, 19]
]
```

```
[[ 16 100  7]
 [ 15   8 17]
 [ 17   1 16]
 [ 17   7 19]]
```

#3 replace phần tử thuộc dòng 1 đến 2, nằm trong cột 3

```
a[1:3,1] = [22,33]
print(a)
```

```
0 1 2
0 [[16 4 7]
1 [15 8 17]
2 [17 1 16]
3 [17 7 19]]
```

```
[[16 4 7]
 [15 22 17]
 [17 33 16]
 [17 7 19]]
```



## 12. Kiểm tra có tồn tại phần tử trong array

```
arr = [  
    [16, 4, 7],  
    [15, 8, 17],  
    [17, 16, 10],  
    [17, 7, 19]  
]  
a = np.array(arr)  
print(16 in a)  
print(900 in a)
```

```
True  
False
```

## 13. Ứng dụng tìm vị trí của phần tử bất kỳ trong mảng

```
for i in range(a.shape[0]):  
    for j in range(a.shape[1]):  
        if a[i, j] == 16:  
            print(i, j)
```

```
0 0  
2 1
```

	0	1	2
0	[[16	4	7]
1	[15	8	17]
2	[17	16	10]
3	[17	7	19]]

