pp-12. Python の numpy

https://www.kkaneko.jp/dblab/intro/pythonintro/index.html

Python を演習と実践で学ぶシリーズ

金子邦彦

pp-12. Python の numpy

URL: https://www.kkaneko.jp/cc/colab/index.html

金子邦彦

配列

要素の並び．要素には**添字**がある．

0	1	2	3	4
8	5	4	1	3

1 次元の配列 [8 5 4 1 3] の**添字**は、

0 1 2 3 4

配列の次元

配列は Python では次のように表示される.

１次元： [要素の並び]

２次元：[[要素の並び] … [要素の並び]]

```
[8  5  4  1  3]
```

１次元

```
[[  1   2   3   4]
 [ 10  20  30  40]
 [100 200 300 400]]
```

２次元

numpy の使用法

import numpy as np

x = np.array([8, 5, 4, 1, 3])

y = np.array([(1, 2, 3, 4), (10, 20, 30, 40), (100, 200, 300, 400)])

「**import numpy as np**」が必要

コンソール 1/A

```
In [1]: import numpy as np
   ...: x = np.array([8, 5, 4, 1, 3])
   ...: y = np.array([(1, 2, 3, 4), (10, 20, 30, 40), (100, 200, 300, 400)])

In [2]: print(x)
[8 5 4 1 3]

In [3]: print(y)
[[  1   2   3   4]
 [ 10  20  30  40]
 [100 200 300 400]]
```

配列の形と次元


```
In [4]: print(x)
[8 5 4 1 3]

In [5]: x.shape
Out[5]: (5,)

In [6]: x.ndim
Out[6]: 1
```

１次元の配列
x を
print(x) で表示

配列の形 は「5」であることを確認

配列の次元 は「1」であることを確認

$shape: 形の取得
ndim: 次元数の取得

配列の形

```
In [7]: print(y)
[[  1   2   3   4]
 [ 10  20  30  40]
 [100 200 300 400]]

In [8]: y.shape
Out[8]: (3, 4)

In [9]: y.ndim
Out[9]: 2
```

shape: 形の取得
ndim: 次元数の取得

２次元の配列 **x** を **print(x)** で表示

配列の形は「3 × 4」であることを確認

配列の次元は「2」であることを確認

numpy の使用例

Python で，**配列**のオブジェクト

a, x を作り，その**形**と**次元数**を表示させる

```
In [66]: a = np.array([8,5,4,1,3])

In [67]: a.shape
Out[67]: (5,)

In [68]: a.ndim
Out[68]: 1

In [69]: print(a)
[8 5 4 1 3]

In [70]: x = np.array([[1,2,3,4], [10,20,30,40], [100,200,300,400]])

In [71]: x.shape
Out[71]: (3, 4)

In [72]: x.ndim
Out[72]: 2

In [73]: print(x)
[[  1   2   3   4]
 [ 10  20  30  40]
 [100 200 300 400]]
```

コンストラクタ（１次元の配列）

- **0要素**　np.zeros(10)

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

- **1要素**　np.ones(10)

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

- **乱数（正規分布）**　np.random.randn(10)

```
[ 1.64670301  1.62923383 -0.01853381 -1.69556792 -0.57932068 -0.00488398
 -0.01879345  0.24965182  1.29970799 -2.27158722]
```

- **要素指定**　np.array([3, 1, 2, 5, 4])

```
[3 1 2 5 4]
```

- **arange による指定** np.arange(-5, 4, 2)

```
[-5 -3 -1  1  3]
```

- **linespace による指定** np.linspace(-2, 2, 9)

```
[-2.  -1.5 -1.  -0.5  0.   0.5  1.   1.5  2. ]
```

コンストラクタ（２次元の配列）

- **0要素**　　np.zeros((2, 3))

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

- **1要素**　　np.ones((2, 3))

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

- **乱数（正規分布）**　　np.random.randn(2, 3)

```
[[ 0.4124597  -1.40007012  0.86764188]
 [-1.95575414  0.92054018  1.87634564]]
```

ravel は，2次元以上の配列を 1次元に変換


```
In [12]: print(y)
[[  1   2   3   4]
 [ 10  20  30  40]
 [100 200 300 400]]

In [13]: print(y.ravel())
[  1   2   3   4  10  20  30  40 100 200 300 400]
```

関連資料

- **Python まとめページ**

https://www.kkaneko.jp/pro/python/googlecolab.html

- **Python の基本**

Google Colaboratory，Paiza.IO を使用.

https://www.kkaneko.jp/cc/colab/index.html

- **Python 入門（全６回）**

Google Colaboratoryを使用.

https://www.kkaneko.jp/cc/pf/index.html

- **Python プログラミング演習（全９回）**

Python Tutor, VisuAlgo を使用

https://www.kkaneko.jp/cc/po/index.html

- **さまざまな Windows アプリケーションのインストールと設定**

https://www.kkaneko.jp/cc/tools/index.html