LẬP TRÌNH PYTHON
NumPy & Pandas

NGUYỄN HẢI TRIỀU[1]

[1]Bộ môn Kỹ thuật phần mềm,

Khoa Công nghệ thông tin, Trường ĐH Nha Trang

NhaTrang, February 2022

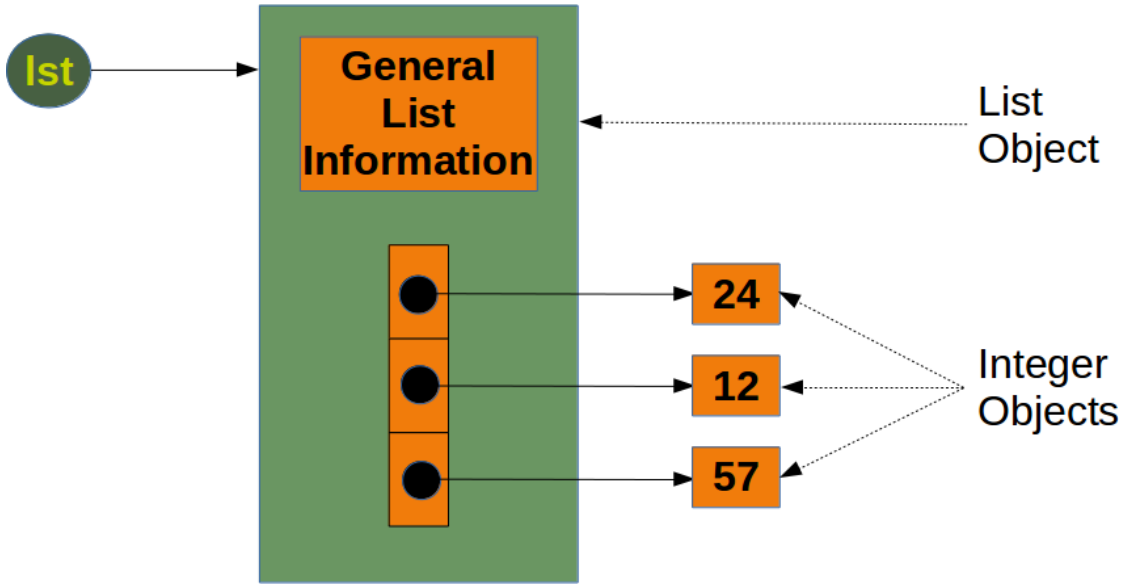
Nội dung

Giới thiệu NumPy

NumPy=Numerical Python

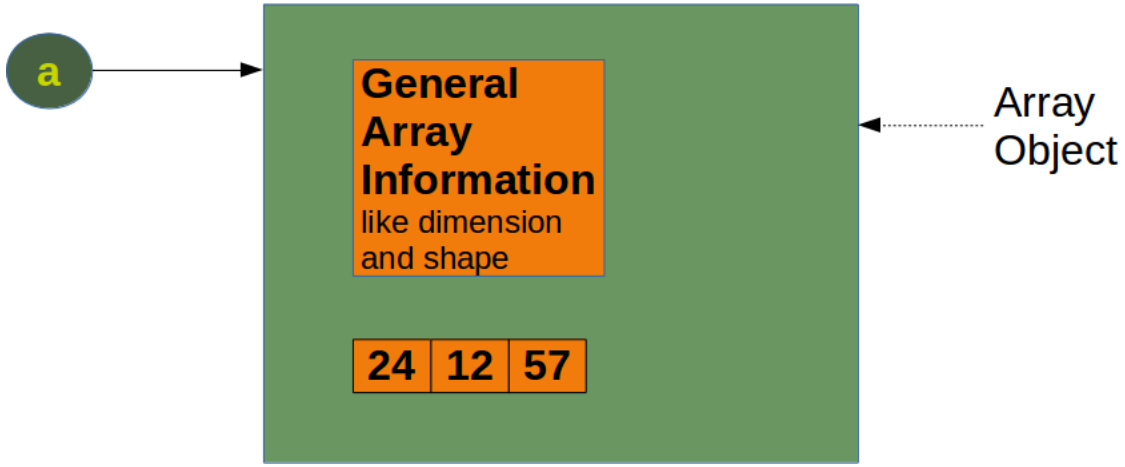
- numpy là một thư viện python, hầu hết được viết bằng C/C++.
- là nền tảng cho ngành **DATA SCIENCE** và **DATA ANALYSIS**
- được sử dụng để làm việc với các mảng, ma trận, đại số tuyến tính
- nó không phải là thư viện cơ bản của Python nên cần phải cài đặt trước khi sử dụng: pip install numpy
- đối tượng mảng trong numpy được gọi **ndarray**

List cũng có chức năng tương tự. Vậy tại sao cần phải sử dụng NumPy? -> nguyên nhân chính là do tốc độ xử lý, sự tiện dụng, dung lượng lưu trữ




```
1 from sys import getsizeof as size
2 lst = [24, 12, 57]
3 size_of_list_object = size(lst)
4 # only green box
5 size_of_elements = len(lst) * size(lst[0]) # 24, 12, 57
6 total_list_size = size_of_list_object + size_of_elements
7 print("Size without the size of the elements: ",
      size_of_list_obje
8 ct)
9 print("Size of all the elements: ", size_of_elements)
10 print("Total size of list, including elements: ", total_list_size
      )
```

Size without the size of the elements: 96. Size of all the elements: 84. Total size of list, including elements: 180




```
1  a = np.array([24, 12, 57])
2  print(size(a)) #-> 120
3  # only green box e
4  e = np.array([])
5  print(size(e)) #-> 96 => kich thuoc 3 so nguyen 120-96=24
6  #########
7  a = np.array([24, 12, 57], np.int8)
8  print(size(a) - 96) # 3
9  a = np.array([24, 12, 57], np.int16)
10 print(size(a) - 96) # 6
11 a = np.array([24, 12, 57], np.int32)
12 print(size(a) - 96) # 12
13 a = np.array([24, 12, 57], np.int64)
14 print(size(a) - 96) # 24
```

```
1  import time
2  import numpy as np
3  size_of_vec = 10000000
4  def pure_python_version():
5      t1 = time.time()
6      X = range(size_of_vec)
7      Y = range(size_of_vec)
8      Z = [X[i] + Y[i] for i in range(len(X)) ]
9      return time.time() - t1
10 def numpy_version():
11     t1 = time.time()
12     X = np.arange(size_of_vec)
13     Y = np.arange(size_of_vec)
14     Z = X + Y
15     return time.time() - t1
16 print("list_version: ",pure_python_version())
17 print("numpy_version: ",numpy_version())
```

Tùy thuộc vào phần cứng của máy nhưng chung quy numpy vẫn nhanh hơn list rất nhiều lần

list_version: 3.264071226119995

numpy_version: 0.2998456949560547

Tạo mảng bằng numpy

Để sử dụng được numpy cần **import** thư viện numpy. Để ngắn gọn cần thêm alias **np**: import numpy as np. Để khởi tạo **ndarray object** với các giá trị cho trước có thể sử dụng phương thức .array(). Ví dụ:

```
1 import numpy as np
2 arr = np.array([1, 2, 3, 4, 5])
3 print(arr) # [1 2 3 4 5]
4 print(type(arr)) # <class 'numpy.ndarray'>
```

Tạo mảng bằng numpy

Khởi tạo mảng với phương thức .arange()

arange([start,] stop[, step], [, dtype=None]). Hàm này trả về các giá trị cách đều nhau trong một khoảng [start, stop) cho trước. Ví dụ:

```
1 import numpy as np
2 a = np.arange(1, 10)# mac dinh step=1
3 print(a) #[1 2 3 4 5 6 7 8 9]
4 x = np.arange(10.4)
5 print(x) # [ 0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
6 x = np.arange(0.5, 10.4, 0.8)
7 print(x) # [ 0.5  1.3  2.1  2.9  3.7  4.5  5.3  6.1  6.9  7.7
      8.5  9.3 10.1]
```

Khởi tạo mảng với phương thức .linspace()

linspace(start, stop, num=50, endpoint=True, retstep=False). Hàm này trả về ndarray có *num* phần tử trong khoảng [start, stop] cho trước.

```
1 import numpy as np
2 print(np.linspace(1,10,10))
3 #[ 1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
4 print(np.linspace(1, 10, 10, endpoint=False))
5 #[1.  1.9 2.8 3.7 4.6 5.5 6.4 7.3 8.2 9.1]
6 print(np.linspace(1, 10, 10, endpoint=False, retstep=True))
7 #return a tuple ('samples', 'step'):
8 #(array([1. , 1.9, 2.8, 3.7, 4.6, 5.5, 6.4, 7.3, 8.2, 9.1]), 0.9)
```

Mảng trong numpy chứa các phần tử cùng kiểu dữ liệu. Để kiểm tra kiểu dữ liệu chung cho mảng, ta sử dụng thuộc tính **.dtype**.

```
1 V = np.array([3.4, 6.9, 99.8, 12.8]) # khoi tao mang 1 chieu
2 print("V: ", V)#V:  [ 3.4  6.9 99.8 12.8]
3 print("Type of V: ", V.dtype)#Type of V:  float64
4 print("Dimension of V: ", np.ndim(V))# Dimension of V:  1
```

Mảng nhiều chiều

NumPy có thể tạo ra mảng n chiều với $n \geq 2$. Chúng ta có thể khởi tạo mảng nhiều chiều dưới dạng truyền các list, tuple lồng vào nhau cho phương thức **.array()**.

```
1 #Vi du khoi tao mang 2 chieu
2 import numpy as np
3 a=np.array([[1,2],[3,4]])
4 print(a,a.ndim) # .ndim dung de lay so chieu cua mang
```

Bài tập: khởi tạo mảng 3 chiều có kích thước $2 \times 2 \times 3$ với các phần tử có giá trị bất kì.

Mảng nhiều chiều

NumPy có thể tạo ra mảng n chiều với $n \geq 2$. Chúng ta có thể khởi tạo mảng nhiều chiều dưới dạng truyền các list, tuple lồng vào nhau cho phương thức **.array()**.

```
1 #Vi du khoi tao mang 2 chieu
2 import numpy as np
3 a=np.array([[1,2],[3,4]])
4 print(a,a.ndim) # .ndim dung de lay so chieu cua mang
```

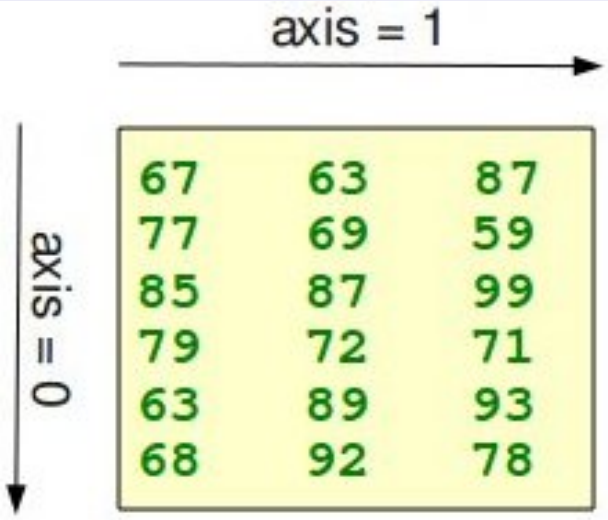
Bài tập: khởi tạo mảng 3 chiều có kích thước $2 \times 2 \times 3$ với các phần tử có giá trị bất kì.

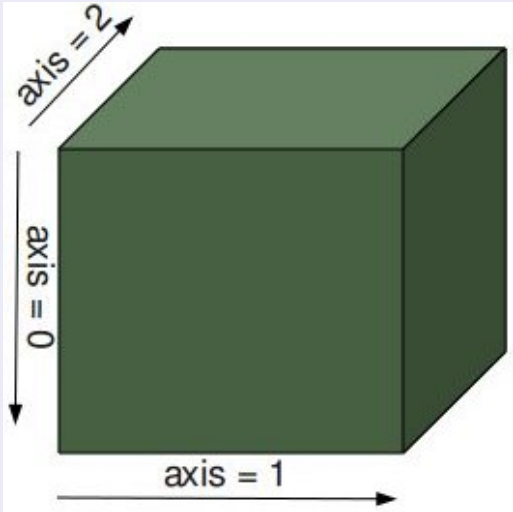
```
1 #Vi du khoi tao mang 3 chieu
2 import numpy as np
3 m133=[1,2,3]
4 m233=[3,4,5]
5 m332=np.array([[m133,m233],[m233,m133]])
6 print(m332, m332.ndim)
```

Shape of an Array

Phương thức .shape()

numpy.shape() trả về kích thước của mảng. Kích thước của mảng là một tuple các số nguyên tương ứng với độ dài các chiều của mảng. Quy ước shape theo các trục:





Shape of an Array

Ví dụ 2.1

```
1 x = np.array([ [67, 63, 87],
2                [77, 69, 59],
3                [85, 87, 99],
4                [79, 72, 71],
5                [63, 89, 93],
6                [68, 92, 78]])
7 print(np.shape(x))#(6,3)
```

Hãy chuyển ma trận $\boldsymbol{x}$ *ở trên thành* $\boldsymbol{x}^T$ *không sử dụng phương thức* ***x.transpose()****?*

Shape of an Array

Ví dụ 2.1

```
1 x = np.array([ [67, 63, 87],
2                [77, 69, 59],
3                [85, 87, 99],
4                [79, 72, 71],
5                [63, 89, 93],
6                [68, 92, 78]])
7 print(np.shape(x))#(6,3)
```

Hãy chuyển ma trận $\boldsymbol{x}$ *ở trên thành* $\boldsymbol{x}^T$ *không sử dụng phương thức* ***x.transpose()****?*

```
1 x = np.array([ [67, 63, 87],
2                [77, 69, 59],
3                [85, 87, 99],
4                [79, 72, 71],
5                [63, 89, 93],
6                [68, 92, 78]])
7 y=np.empty((3,6))
8 for i in range(np.shape(x)[0]):
9     for j in range(np.shape(x)[1]):
10        y[j][i]=x[i][j]
11 print(y)
```

khởi tạo các mảng đặc biệt

- mảng khởi tạo chứa phần tử có giá trị 0: **np.zeros((2,4))**
- mảng khởi tạo chứa phần tử có giá trị 1: **np.ones((2,4))**
- mảng khởi tạo chứa phần tử có giá trị ngẫu nhiên: **np.empty((2,4))**
- mảng khởi tạo chứa phần tử có giá trị cho trước: **np.full((2,2), 3)**
- khởi tạo ma trận đường chéo chính: **np.eye(3,3)**

Chỉ số mảng và slicing

Chỉ số mảng trong NumPy

Các phần tử trong mảng của NumPy cũng được truy cập thông qua chỉ số giống như list, tuple. Ví dụ:

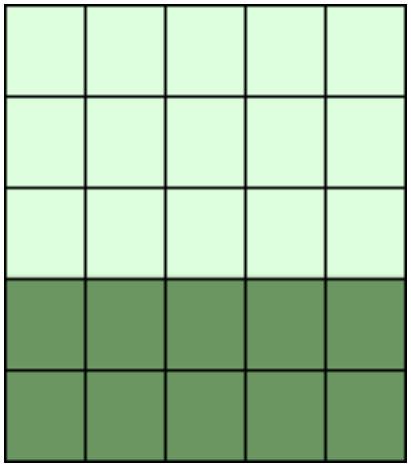
```
1 F = np.array([1, 1, 2, 3, 5, 8, 13, 21])
2 print(F[0])# print the first element of F->1
3 print(F[-1])# print the last element of F->21
4 A = np.array([ [3.4, 8.7, 9.9], [1.1, -7.8, -0.7], [4.1, 12.3, 4.
      8]])
5 print(A[1][0])#1.1
6 #usisng only one pair of square brackets and all the indices are
      separated by commas:
7 print(A[-1,-1])#4.8
```

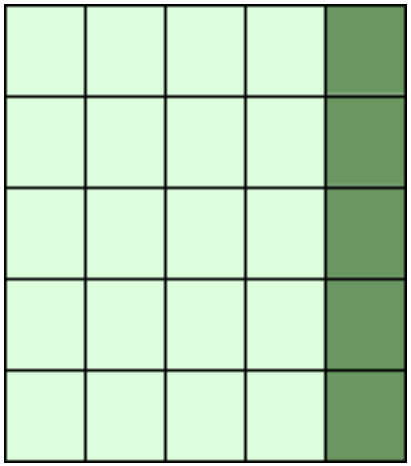
NumPy Array Slicing

Cắt mảng là lấy các phần tử trong mảng từ một đoạn vị trí trong mảng gốc. Cú pháp tổng quát: $A[start : stop : step]$ cho mảng 1 chiều và có thể áp dụng cho mảng nhiều chiều.

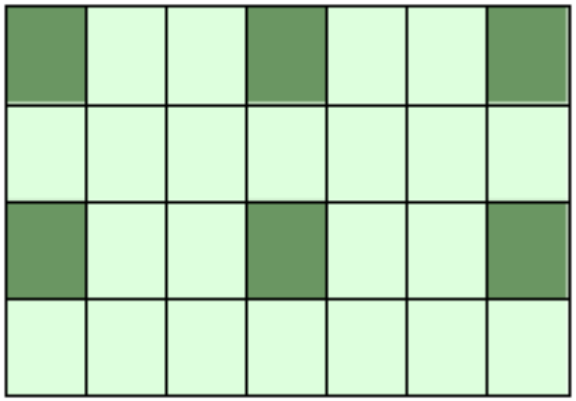
```
S = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
print(S[2:5])#[2 3 4]
print(S[:4])#[0 1 2 3]
print(S[6:])#[6 7 8 9]
print(S[:])#[0 1 2 3 4 5 6 7 8 9]
##for multidimensional
A = np.array([
[11, 12, 13, 14, 15],
[21, 22, 23, 24, 25],
[31, 32, 33, 34, 35],
[41, 42, 43, 44, 45],
[51, 52, 53, 54, 55]])
print(A[:3, 2:])
```

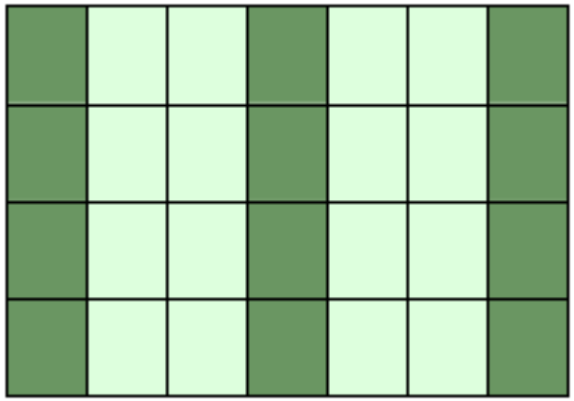
```
1 A = np.array([
2 [11, 12, 13, 14, 15],
3 [21, 22, 23, 24, 25],
4 [31, 32, 33, 34, 35],
5 [41, 42, 43, 44, 45],
6 [51, 52, 53, 54, 55]])
7 print(A[3:, :])
8 print(A[:, 4:])
```





```
X = np.arange(28).reshape(4, 7)
print(X)
#[[ 0  1  2  3  4  5  6]
 [ 7  8  9 10 11 12 13]
 [14 15 16 17 18 19 20]
 [21 22 23 24 25 26 27]]
print(X[::2, ::3])
print(X[::, ::3])
```





Lưu ý

Khi cắt một mảng trong numpy gán vào một đối tượng mới, thì mọi thay đổi trong mảng mới cũng sẽ thay đổi ở mảng gốc. Ví dụ:

```
A = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
S = A[2:6]
S[0] = 22
S[1] = 23
print(A)
#[0  1  22  23  4  5  6  7  8  9]
```

Tuy nhiên điều này không đúng cho list, tuple.

```
lst = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
lst2 = lst[2:6]
lst2[0] = 22
lst2[1] = 23
print(lst)
#[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Numpy Data Objects, dtype

Các kiểu dữ liệu trong NumPy

Danh sách tất cả các kiểu dữ liệu trong NumPy: i - integer; b - boolean; u - unsigned integer; f - float; c - complex float; m - timedelta; M - datetime; O - object; S - string; U - unicode string; V - fixed chunk of memory for other type (void).

dtype

Để kiểm tra kiểu dữ liệu của một mảng, sử dụng thuộc tính kiểu dữ liệu dtype trong lớp **numpy.dtype**. Kiểu dữ liệu của mảng phụ thuộc vào kiểu dữ liệu lớn nhất xuất hiện trong mảng.

```
1 import numpy as np
2 a = np.array([[1, 2],[3, 4.5]])
3 print(a.dtype) #float64
```

Tạo mảng với kiểu dữ liệu xác định

Với tùy chọn đối số đầu vào dtype= trong phương thước khởi tạo mảng np.array(), chúng ta có thể tạo được mảng với kiểu dữ liệu quy định định trước.

```
1  import numpy as np
2  i16 = np.dtype(np.int16)# dinh nghia KDL moi dua tren KDL co so
3  lst = [ [3.4, 8.7, 9.9],
4          [1.1, -7.8, -0.7],
5          [4.1, 12.3, 4.8] ]
6  A = np.array(lst)
7  B = np.array(lst,dtype='i')
8  C = np.array(lst, dtype=i16)
9  D = np.array(lst, dtype='i4')
10 print(A.dtype, B.dtype, C.dtype, D.dtype)
11 #float64 int32 int16 int4
```

Structured Arrays

Tạo mảng có cấu trúc

Làm thế nào để tạo một mảng có cấu trúc với kiểu dữ liệu quy định trước như bảng sau

	A	B	C	D
1	Country	Density	Area	Population
2	Netherlands	393	41526	16928800
3	Belgium	337	30510	11007020
4	United Kingdom	256	243610	62262000
5	Germany	233	357021	81799600
6	Liechtenstein	205	160	32842
7	Italy	192	301230	59715625
8	Switzerland	177	41290	7301994
9	Luxembourg	173	2586	512000
10	France	111	547030	63601002
11	Austria	97	83858	8169929
12	Greece	81	131940	11606813
13	Ireland	65	70280	4581269
14	Sweden	20	449964	9515744
15	Finland	16	338424	5410233
16	Norway	13	385252	5033675

dtype cho phép xác định các kiểu dữ liệu riêng biệt cho mỗi cột: lần lượt tạo mảng có cấu trúc cho từng cột. KDL được định nghĩa cho cả mảng: np.dtype([('tên_cột$_0$', KDL),('tên_cột$_1$', KDL),...]).

```
1  dt = np.dtype([('country', 'S20'), ('density', 'i4'), ('area', '
       i4'), ('population', 'i4')])
2  population_table = np.array([
3      ('Netherlands', 393, 41526, 16928800),
4      ('Belgium', 337, 30510, 11007020),
5      ('United Kingdom', 256, 243610, 62262000),
6      ('Germany', 233, 357021, 81799600),
7      ('Liechtenstein', 205, 160, 32842),
8      ('Italy', 192, 301230, 59715625),
9      ('Switzerland', 177, 41290, 7301994),
10     ('Luxembourg', 173, 2586, 512000),
11     ('France', 111, 547030, 63601002),
12     ('Austria', 97, 83858, 8169929),
13     ('Greece', 81, 131940, 11606813),
14     ('Ireland', 65, 70280, 4581269),
15     ('Sweden', 20, 449964, 9515744),
16     ('Finland', 16, 338424, 5410233),
17     ('Norway', 13, 385252, 5033675)],
18     dtype=dt)
```

Truy cập các phần tử trong mảng có cấu trúc

```
#in ra 4 hang dau tien
print(population_table[:4])
#[(b'Netherlands', 393,  41526, 16928800)
 (b'Belgium', 337,  30510, 11007020)
 (b'United Kingdom', 256, 243610, 62262000)
 (b'Germany', 233, 357021, 81799600)]
# Truy cap tung cot
print(population_table['density'])
#[393 337 256 233 205 192 177 173 111  97  81  65  20  16  13]
print(population_table['area'][2:5])
#[243610 357021 160]
```

Mảng kí tự Unicode

Vấn đề trong mảng chứa kí tự Unicode

```
import numpy as np
dt = np.dtype([('country', 'S20'), ('density', 'i4'), ('area', 'i4'),
('population', 'i4')])
population_table = np.array([('Việt Nam', 233, 357021,
81799600)],dtype=dt)
```

-> UnicodeEncodeError: 'ascii' codec can't encode character 'ỉec7' in position 2: ordinal not in range(128)

Giải pháp

```
dt = np.dtype([('country', np.compat.unicode, 20), ('density',
'i4'), ('area', 'i4'), ('population', 'i4')])
population_table = np.array([('Việt Nam', 233, 357021,
81799600)],dtype=dt)
```

Lưu mảng có cấu trúc

Ghi File

Trong hầu hết các ứng dụng, cần phải lưu dữ liệu mảng vào file. Để ghi dữ liệu mảng có cấu trúc như trên vào file, ta sử dụng lệnh np.savetxt(). Ví dụ:

```
1  import numpy as np
2  dt = np.dtype([('country', np.compat.unicode, 20),
3                 ('density', 'i4'),
4                 ('area', 'i4'),
5                 ('population', 'i4')])
6  population_table = np.array([
7      ('Netherlands', 393, 41526, 16928800),
8      ('Belgium', 337, 30510, 11007020),
9      ('United Kingdom', 256, 243610, 62262000),
10     ('Germany', 233, 357021, 81799600),
11     ('Liechtenstein', 205, 160, 32842),
12     ('Sweden', 20, 449964, 9515744)],
13     dtype=dt)
14 np.savetxt('population_table.csv',population_table,fmt="%s;%d;%d
       ;%d",delimiter=";")
```

Đọc file dữ liệu

Đọc File

Để đọc file dữ liệu đã lưu ở trên, ta sử dụng lệnh np.genfromtxt() của lớp Numpy. Ví dụ:

```
1 import numpy as np
2 dt = np.dtype([('country', np.compat.unicode, 20), ('density', '
      i4'), ('area', 'i4'), ('population', 'i4')])
3 x = np.genfromtxt("./population_table.csv", dtype=dt, delimiter="
      ;")
4 print(x[:3])
```

Hoặc dùng phương thức np.loadtxt()

```
1 import numpy as np
2 dt = np.dtype([('country', np.compat.unicode, 20), ('density', '
      i4'), ('area', 'i4'), ('population', 'i4')])
3 y = np.loadtxt("population_table.csv", dtype=dt,delimiter=";")
4 print(y)
```

Bài tập

Ví dụ 2.2

Xây dựng một mảng có cấu trúc với 2 cột: tên sản phẩm và giá của sản phẩm. Thực hiện các yêu cầu sau:

1. *ghi mảng đó thành file .txt vào thư mục data*
2. *đọc file vừa ghi*
3. *lọc và hiển thị ra những sản phẩm có giá lớn hơn 10000*
4. *lọc và hiển thị ra những sản phẩm kí tự bắt đầu bằng các kí tự 'b', 'k' trong tên*

Tài liệu tham khảo

Trung tâm tin học , Đại Học KHTN Tp.HCM
Lập trình Python nâng cao. *03/2017*.

Bernd Klein
Data Analysis: Numpy, Matplotlib and Pandas.
bernd.klein@python-course.eu, 2021.

Luciano Ramalho
Fluent Python (2nd Edition). *O'Reilly Media, Inc, 2021*.

Python Software Foundation
`https://docs.python.org/3/tutorial/`

Temporary page!

LaTeX was unable to guess the total number of pages cor
there was some unprocessed data that should have been
the final page this extra page has been added to receive i
If you rerun the document (without altering it) this surp
will go away, because LaTeX now knows how many pages
for this document.