

▼ Chapter 3 - Exercise 1 a:

▼ Các kiến thức sử dụng trong bài tập:

Tạo mảng (numpy array):

1. Tạo mảng có giá trị trong khoảng từ a đến b - 1:

```
biến_mảng = np.arange(a, b)
```

2. Tạo mảng từ 1 dãy số cho trước:

```
biến_mảng = np.array([dãy số])
```

3. Tạo mảng theo điều kiện từ mảng đã có:

```
biến_mảng_mới = biến_mảng[<điều kiện>]
```

```
ví dụ: arr_even = arr[arr % 2 == 0]
```

Các thao tác trên mảng:

Xem số phần tử có trong mảng: `biến_mảng.shape`

▼ Thực hiện các yêu cầu sau và đối chiếu với kết quả cho trước:

```
[ ] import numpy as np
```

```
[ ] # Câu 1: Tạo numpy array có giá trị từ 0-9 và lưu vào biến arr
```

```
arr = np.arange(0, 10)
```

```
# Hiển thị các phần tử có trong arr  
print(arr)
```

```
# Xem kiểu dữ liệu (type) của arr  
print(type(arr))
```

```
# Xem kích thước (shape) của arr  
print(arr.shape)
```

```
[0 1 2 3 4 5 6 7 8 9]  
<class 'numpy.ndarray'>  
(10,)
```

```
[ ] # Câu 2: Từ array arr ở câu 1 => tạo arr_odd và arr_even
arr_odd = arr[arr%2!=0]
arr_even = arr[arr%2==0]

# Hiển thị các phần tử có trong arr_odd và arr_even
print('Odd array: \n', arr_odd)
print('Even array: \n', arr_even)
```

```
Odd array:
[1 3 5 7 9]
Even array:
[0 2 4 6 8]
```



```
# Câu 3: Từ array arr ở câu 1 => tạo arr_update_1 với các phần tử chẵn giữ nguyên, các phần tử lẻ thay bằng 100
arr_update_1 = np.where(arr%2==0, arr, 100)

# Hiển thị các phần tử có trong arr_update_1
print(arr_update_1)
```



```
[ 0 100  2 100  4 100  6 100  8 100]
```

Chapter 3 - Exercise 1b:

Các kiến thức sử dụng trong bài tập:

Các thao tác trên mảng:

1. Lấy các phần tử (không trùng) xuất hiện trong cả 2 mảng a và b: hàm **np.intersect1d**
2. Lấy các phần tử chỉ xuất hiện trong mảng a và không có trong mảng b: hàm **np.setdiff1d**

Thực hiện các yêu cầu sau và đối chiếu với kết quả cho trước:

```
[ ] import numpy as np
```

```
[ ] # Câu 1: Cho 2 array arr_a = [1,2,3,2,3,4,3,4,5,6] và arr_b = [7,2,10,2,7,4,9,4,9,8]
    # Tạo array mới arr_c chỉ lấy duy nhất các phần tử xuất hiện ở cả array arr_a và array arr_b

    arr_a = np.array([1,2,3,2,3,4,3,4,5,6])
    arr_b = np.array([7,2,10,2,7,4,9,4,9,8])

    arr_c = np.intersect1d(arr_a, arr_b)
    print(arr_c)
```

[2 4]



Câu 2: Từ 2 array arr_a và arr_b ở câu 1 => Tạo array mới arr_d chứa các phần tử chỉ xuất hiện ở array arr_a

```
arr_d = np.setdiff1d(arr_a, arr_b)
print(arr_d)
```



[1 3 5 6]

► Nhấn vào đây để xem kết quả !

```
[ ] # Câu 3: Cho array arr_e = [2, 6, 1, 9, 10, 3, 27, 8, 6, 25, 16]
    # Tạo array arr_f chỉ chứa các phần tử có giá trị từ 5 đến 10 của arr_e

arr_e = np.array([2, 6, 1, 9, 10, 3, 27, 8, 6, 25, 16] )
arr_f = arr_e[(arr_e >=5) * (arr_e <= 10)]
print(arr_f)
```

[6 9 10 8 6]

Chapter 3 - Exercise 1c:

Các kiến thức sử dụng trong bài tập:

Tạo mảng (numpy array):

1. Tạo mảng là 1 dãy số bắt đầu từ a đến b - 1: hàm **arange**
2. Tạo mảng từ 1 dãy số cho trước
3. Tạo mảng với các phần tử là số 0: hàm **zeros**
4. Tạo mảng theo điều kiện từ mảng đã có

Các thao tác trên mảng:

1. Thêm phần tử vào cuối mảng đang có: hàm **append**
2. Chèn thêm phần tử vào vị trí bất kỳ trong mảng đang có: hàm **insert**
3. Xóa phần tử trong mảng: hàm **delete**

Thực hiện các yêu cầu sau và đối chiếu với kết quả cho trước:

```
[ ] import numpy as np
```

```
[ ] # Câu 1: Tạo array arr_zeros có 10 phần tử 0, cập nhật phần tử ở vị trí thứ 5 là 1
arr_zeros = np.zeros(10)
print(arr_zeros)

arr_zeros[4] = 1
print(arr_zeros)
```

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
[0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
```

► Nhấn vào đây để xem kết quả !

```
[ ] # Câu 2: Tạo và in array arr_h có giá trị từ 10 đến 24
arr_h = np.arange(10, 25)
print(arr_h)

# In danh sách các phần tử theo thứ tự đảo ngược của arr_h vừa tạo
print(arr_h[::-1])
```

```
[10 11 12 13 14 15 16 17 18 19 20 21 22 23 24]
[24 23 22 21 20 19 18 17 16 15 14 13 12 11 10]
```



```
[ ] # Câu 3: Cho array arr_k = [1, 2, 0, 8, 2, 0, 1, 3, 0, 5, 0]
# Tạo array arr_l từ arr_k với các phần tử khác 0

arr_k = np.array([1, 2, 0, 8, 2, 0, 1, 3, 0, 5, 0] )
print(arr_k)

arr_l = arr_k[arr_k != 0]
print(arr_l)
```

```
[1 2 0 8 2 0 1 3 0 5 0]
[1 2 8 2 1 3 5]
```

► Nhấn vào đây để xem kết quả !

```
[ ] # Câu 4: Từ array arr_l của câu 3, thêm 2 phần tử có giá trị là 10 và 20 vào cuối array
arr_l = np.append(arr_l, [10, 20])
print(arr_l)
```

```
[ 1  2  8  2  1  3  5 10 20]
```

```
[ ] # Câu 5: Từ array của câu 4, thêm phần tử có giá trị 100 vào vị trí có index = 5

arr_1 = np.insert(arr_1, 5, 100)
print(arr_1)
```

```
[ 1  2  8  2  1 100  3  5 10 20]
```

► Nhấn vào đây để xem kết quả !

```
[ ] # Câu 6: Từ array của câu 5, xóa các phần tử tại vị trí có index = 0, 1, 2

arr_1 = np.delete(arr_1, [0, 1, 2])
print(arr_1)
```

```
[ 2  1 100  3  5 10 20]
```

Chapter 3 - Exercise 2: Đọc và chuyển dữ liệu, sau đó tính BMI theo điều kiện, truy xuất dữ liệu

Dữ liệu được trích xuất từ http://wiki.stat.ucla.edu/socr/index.php/SOCR_Data_MLB_HeightsWeights

Ghi chú: Major League Baseball (MLB) là giải đấu bóng chày chuyên nghiệp. Major League Baseball có tổng cộng 30 đội bóng đến từ nhiều bang khác nhau của Mỹ và Canada (29 đội từ Mỹ và 1 đội từ Canada). MLB luôn được sự quan tâm lớn của hầu hết fan bóng chày trên toàn thế giới, và cũng được xem là giải đấu nổi tiếng và uy tín nhất, tập hợp những cầu thủ có trình độ cao nhất trong bộ môn này. Dữ liệu **heights** (tính theo inches) và **weights** (tính theo pounds) là chiều cao và cân nặng của các cầu thủ có tham gia 1 số giải của MLB.

Cho tập tin dữ liệu heights_1.txt, weights_1.txt

Các kiến thức sử dụng trong bài tập:

1. Sử dụng các phép toán số học trên mảng
2. Truy xuất phần tử của mảng thông qua chỉ số

Thực hiện các yêu cầu sau, và đối chiếu với kết quả cho trước:

```
[ ] import numpy as np
```

```
[ ] # Đọc file
with open('data\heights_1.txt', 'r') as f:
    x = f.read()
x=x.replace('[', '')
x=x.replace(']', '')
height = x.split(sep=',')
height
arr_height = np.asarray(height, dtype=int)
arr_height

array([74, 74, 72, ..., 75, 75, 73])
```

```
[ ] # Chép dữ liệu từ tập tin heights_1.txt vào list height
height = [74, 74, 72, 72, 73, 69, 69, 71, 76, 71, 73, 73, 74, 74, 69, 70, 73, 75, 78, 79, 76, 74, 76, 72, 71, 75, 77,
# Chép dữ liệu từ tập tin weights_1.txt vào list weight
weight = [180, 215, 210, 210, 188, 176, 209, 200, 231, 180, 188, 180, 185, 160, 180, 185, 189, 185, 219, 230, 205, 23
```

```
[ ] print(len(height))
print(len(weight))
```

```
1015
1015
```

```
[ ] # Câu 1: Tạo numpy array arr_height từ list height
arr_height = np.asarray(height)
# In danh sách các phần tử của arr_height
print(arr_height)
# Cho biết kích thước (shape) của arr_height
print('Kích thước arr_height = ', arr_height.shape)
```

```
[74 74 72 ... 75 75 73]
Kích thước arr_height = (1015,)
```

► Nhấn vào đây để xem kết quả !

```
[ ] # Câu 2: Tạo numpy array arr_weight từ list weight
arr_weight = np.asarray(weight)
# In danh sách các phần tử của arr_weight
print(arr_weight)
# Cho biết kích thước (shape) của arr_weight
print(arr_weight.shape)
```

```
[180 215 210 ... 205 190 195]
(1015,)
```

```
[ ] # Câu 3: Cho hệ số quy đổi từ inch sang m là 0.0254
# Tạo array arr_height_m dựa trên công thức: arr_height * hệ số quy đổi
arr_height_m = arr_height * .0254
# In danh sách các phần tử của arr_height_m
print(arr_height_m)
```

```
[1.8796 1.8796 1.8288 ... 1.905 1.905 1.8542]
```

► Nhấn vào đây để xem kết quả !

```
[ ] # Câu 4: Cho hệ số quy đổi từ pound sang kg là 0.453592
# Tạo array arr_weight_kg dựa trên công thức: arr_weight * hệ số quy đổi
arr_weight_kg = arr_weight * 0.453592
# In danh sách các phần tử của arr_weight_kg
print(arr_weight_kg)
```

```
[81.64656 97.52228 95.25432 ... 92.98636 86.18248 88.45044]
```

► Nhấn vào đây để xem kết quả !

```
[ ] # Câu 5: Tính giá trị BMI (Body Mass Index) của arr_height_m và arr_weight_kg và lưu vào arr_bmi
# Gợi ý: Tính theo công thức BMI = Cân nặng / (Chiều cao * Chiều cao)
arr_bmi = arr_weight_kg/(arr_height_m**2)
# In ra danh sách các phần tử của arr_bmi
print(arr_bmi)
```

```
[23.11037639 27.60406069 28.48080465 ... 25.62295933 23.74810865
25.72686361]
```



```
[ ] # Câu 6: Cho biết giá trị cân nặng ở vị trí index = 50 trong arr_weight_kg
print(arr_weight_kg[50])
```

90.7184

► Nhấn vào đây để xem kết quả !


```
[ ] # Câu 7: Tạo array arr_height_m_100 bao gồm các phần tử có vị trí index từ 100 đến 110 (lấy cả index 110) trong arr_h
arr_height_m_100 = arr_height_m[100:111]
print(arr_height_m_100)
```

[1.8542 1.8796 1.8288 1.8542 1.7526 1.8288 1.8542 1.905 1.905 1.8542
1.8288]

► Nhấn vào đây để xem kết quả !


```
[ ] # Câu 8: Tạo và in kết quả của biểu thức điều kiện dùng để lấy ra các cầu thủ bóng chày có bmi < 21
bmi_less_21 = arr_bmi < 21
print(bmi_less_21)
# Áp dụng biểu thức điều kiện đã tạo để in ra các cầu thủ bóng chày có bmi < 21 trong arr_bmi
print(arr_bmi[bmi_less_21])
```

[False False False ... False False False]
[20.54255679 20.54255679 20.69282047 20.69282047 20.34343189 20.34343189
20.69282047 20.15883472 19.4984471 20.69282047 20.9205219]

 # Câu 9: Cho biết chiều cao trung bình và cân nặng trung bình của các cầu thủ

```
avg_height = np.mean(arr_height_m)
print('Chiều cao trung bình:', avg_height)
```

```
avg_weight = np.mean(arr_weight_kg)
print('Cân nặng trung bình: ', avg_weight)
```

 Chiều cao trung bình: 1.8717172413793102
Cân nặng trung bình: 91.33019058916256

[] # Câu 10: Cho biết chiều cao và cân nặng lớn nhất của các cầu thủ

```
max_height = np.max(arr_height_m)
print('Chiều cao lớn nhất: ', max_height)
```

```
max_weight = np.max(arr_weight_kg)
print('Cân nặng lớn nhất: ', max_weight)
```

Chiều cao lớn nhất: 2.1082
Cân nặng lớn nhất: 131.54167999999999

[] # Câu 11: Cho biết chiều cao và cân nặng nhỏ nhất của các cầu thủ

```
min_height = np.min(arr_height_m)
print('Chiều cao thấp nhất: ', min_height)
```

```
min_weight = np.min(arr_weight_kg)
print('Cân nặng bé nhất: ', min_weight)
```

Chiều cao thấp nhất: 1.7018
Cân nặng bé nhất: 68.0388

Chapter 3 - Exercise 3

Các kiến thức sử dụng trong bài tập:

Các xử lý trên mảng 2 chiều

1. Biến đổi chiều của mảng (vd; mảng 1 chiều → mảng 2 chiều): **reshape**
2. Tạo mảng với tất cả các phần tử có giá trị là 1
3. Thay đổi thứ tự các dòng trong mảng
4. Thay đổi thứ tự các cột trong mảng
5. Đảo ngược các dòng trong mảng
6. Đảo ngược các cột trong mảng
7. Kiểm tra xem trong mảng có phần tử Nan (Null) hay không: **isnan**

Thực hiện các yêu cầu sau đây và đối chiếu với kết quả cho trước:

```
[ ] import numpy as np
```

```
[ ] # Câu 1: Tạo array arr có kích thước 3x3 với tất cả các giá trị đều là True.  
    array_arr=np.ones([3,3],dtype=bool)
```

```
# Hiển thị các phần tử của array arr  
print(array_arr)
```

```
[[ True  True  True]  
 [ True  True  True]  
 [ True  True  True]]
```

```
[ ] # Câu 2: Cho array arr_1D = np.array([0 1 2 3 4 5 6 7 8]).
arr_1D = np.arange(0,9)
print('Arr_1D: ',arr_1D)
# Hãy chuyển thành array 2 chiều có kích thước 3x3 và lưu vào arr_2D.
arr_2D=arr_1D.reshape(3,3)
# Xem danh sách các phần tử của arr_2D.
print('Original arr_2D\n',arr_2D)

# Trong arr_2D. Chuyển cột 1 sang cột 3 và ngược lại. => Xem lại danh sách các phần tử của arr_2D
print('New arr_2D')
arr_2D[:,[0,2]]=arr_2D[:,[2,0]]
print(arr_2D)
```

Arr_1D: [0 1 2 3 4 5 6 7 8]

Original arr_2D

[[0 1 2]

[3 4 5]

[6 7 8]]

New arr_2D

[[2 1 0]

[5 4 3]

[8 7 6]]

```
[ ] # Câu 3: Sử dụng array arr_2D của câu 2 (sau khi đổi thứ tự cột), chuyển dòng 1 sang dòng 2 và ngược lại  
arr_2D[[0,1],:]=arr_2D[[1,0],:]  
  
# Xem lại danh sách các phần tử của arr_2D  
print(arr_2D)
```

```
[[5 4 3]  
 [2 1 0]  
 [8 7 6]]
```

► Nhấn vào đây để xem kết quả !

```
[ ] # Câu 4: Sử dụng array arr_2D của câu 3, Đảo ngược các dòng của array arr_2D  
arr_2D[[0,1,2],:]=arr_2D[[2,1,0],:]  
# Xem lại danh sách các phần tử của arr_2D  
print(arr_2D)
```

```
[[8 7 6]  
 [2 1 0]  
 [5 4 3]]
```

```
[ ] # Câu 5: Sử dụng array arr_2D của câu 4, Đảo ngược các cột của array arr_2D => Xem lại danh sách các phần tử của arr_2D
arr_2D[:, [0,1,2]] = arr_2D[:, [2,1,0]]
print(arr_2D)
```

```
[[6 7 8]
 [0 1 2]
 [3 4 5]]
```

► Nhấn vào đây để xem kết quả !

```
[ ] # Câu 6: Cho arr_2D_null = np.array([[1, 2, 3], [np.NaN, 5, 6], [7, np.NaN, 9], [4, 5, 6]]),
# Kiểm tra xem trong array có giá trị rỗng nào không?
```

```
arr_2D_null = np.array([[1, 2, 3], [np.NaN, 5, 6], [7, np.NaN, 9], [4, 5, 6]])
print(arr_2D_null)
```

```
arr_nan = np.sum(arr_2D_null)
```

```
print('Have nan?: ', np.isnan(arr_nan))
```

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

```
Have nan?: True
```

```
[ ] # Câu 7: Sử dụng array arr_2D_null của câu 6, thay thế giá trị null bằng 0 => Xem lại danh sách các phần tử của arr_2
print('Before:\n',arr_2D_null)

arr_2D_null[np.isnan(arr_2D_null)]=0
print('After:\n',arr_2D_null)
```

Before:

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

After:

```
[[1. 2. 3.]
 [0. 5. 6.]
 [7. 0. 9.]
 [4. 5. 6.]]
```

Chapter 3 - Exercise 4: Thao tác dữ liệu mảng dữ liệu baseball

Các kiến thức sử dụng trong bài tập:

Các xử lý trên mảng 2 chiều

1. Kiểm tra kích thước của mảng
2. Truy xuất dòng bất kỳ trong mảng
3. Truy xuất cột bất kỳ trong mảng

Cho tập tin baseball_2D.txt => chép dữ liệu từ tập tin vào list là baseball, sau đó thực hiện các yêu cầu sau, và đối chiếu với kết quả được cung cấp:

Dữ liệu baseball cho biết chiều cao (cột 1) tính theo inch và cân nặng (cột 2) tính theo pounds của các cầu thủ


```
[ ] import numpy as np
```

```
[ ] # dữ liệu baseball
```

```
baseball = [[74, 180], [74, 215], [72, 210], [72, 210], [73, 188], [69, 176], [69, 209], [71, 200], [76, 231],
[71, 180], [73, 188], [73, 180], [74, 185], [74, 160], [69, 180], [70, 185], [73, 189], [75, 185], [78, 219],
[79, 230], [76, 205], [74, 230], [76, 195], [72, 180], [71, 192], [75, 225], [77, 203], [74, 195], [73, 182],
[74, 188], [78, 200], [73, 180], [75, 200], [73, 200], [75, 245], [75, 240], [74, 215], [69, 185], [71, 175],
[74, 199], [73, 200], [73, 215], [76, 200], [74, 205], [74, 206], [70, 186], [72, 188], [77, 220], [74, 210],
[70, 195], [73, 200], [75, 200], [76, 212], [76, 224], [78, 210], [74, 205], [74, 220], [76, 195], [77, 200],
[81, 260], [78, 228], [75, 270], [77, 200], [75, 210], [76, 190], [74, 220], [72, 180], [72, 205], [75, 210],
[73, 220], [73, 211], [73, 200], [70, 180], [70, 190], [70, 170], [76, 230], [68, 155], [71, 185], [72, 185],
[75, 200], [75, 225], [75, 225], [75, 220], [68, 160], [74, 205], [78, 235], [71, 250], [73, 210], [76, 190],
[74, 160], [74, 200], [79, 205], [75, 222], [73, 195], [76, 205], [74, 220], [74, 220], [73, 170], [72, 185],
[74, 195], [73, 220], [74, 230], [72, 180], [73, 220], [69, 180], [72, 180], [73, 170], [75, 210], [75, 215],
[73, 200], [72, 213], [72, 180], [76, 192], [74, 235], [72, 185], [77, 235], [74, 210], [77, 222], [75, 210],
[76, 230], [80, 220], [74, 180], [74, 190], [75, 200], [78, 210], [73, 194], [73, 180], [74, 190], [75, 240],
[76, 200], [71, 198], [73, 200], [74, 195], [76, 210], [76, 220], [74, 190], [73, 210], [74, 225], [70, 180],
[72, 185], [73, 170], [73, 185], [73, 185], [73, 180], [71, 178], [74, 175], [74, 200], [72, 204], [74, 211],
[71, 190], [74, 210], [73, 190], [75, 190], [75, 185], [79, 290], [73, 175], [75, 185], [76, 200], [74, 220],
[76, 170], [78, 220], [74, 190], [76, 220], [72, 205], [74, 200], [76, 250], [74, 225], [75, 215], [78, 210],
[75, 215], [72, 195], [74, 200], [72, 194], [74, 220], [70, 180], [71, 180], [70, 170], [75, 195], [71, 180],
[71, 170], [73, 206], [72, 205], [71, 200], [73, 225], [72, 201], [75, 225], [74, 233], [74, 180], [75, 225],
[73, 180], [77, 220], [73, 180], [76, 237], [75, 215], [74, 190], [76, 235], [75, 190], [73, 180], [71, 165],
[76, 195], [75, 200], [72, 190], [71, 190], [77, 185], [73, 185], [74, 205], [71, 190], [72, 205], [74, 206],
[75, 220], [73, 208], [72, 170], [75, 195], [75, 210], [74, 190], [72, 211], [74, 230], [71, 170], [70, 185],
[74, 185], [77, 241], [77, 225], [75, 210], [75, 175], [78, 230], [75, 200], [76, 215], [73, 198], [75, 226],
[75, 278], [79, 215], [77, 230], [76, 240], [71, 184], [75, 219], [74, 170], [69, 218], [71, 190], [76, 225],
[72, 220], [72, 176], [70, 190], [72, 197], [73, 204], [71, 167], [72, 180], [71, 195], [73, 220], [72, 215],
[73, 185], [74, 190], [74, 205], [72, 205], [75, 200], [74, 210], [74, 215], [77, 200], [75, 205], [73, 211],
[72, 190], [71, 208], [74, 200], [77, 210], [75, 232], [75, 230], [75, 210], [78, 220], [78, 210], [74, 202],
[76, 212], [78, 225], [76, 170], [70, 190], [72, 200], [80, 237], [74, 220], [74, 170], [71, 193], [70, 190],
[72, 150], [71, 220], [74, 200], [71, 190], [72, 185], [71, 185], [74, 200], [69, 172], [76, 220], [75, 225],
[75, 190], [76, 195], [73, 219], [76, 190], [73, 197], [77, 200], [73, 195], [72, 210], [72, 177], [77, 220],
[77, 235], [71, 180], [74, 195], [74, 195], [73, 190], [78, 230], [75, 190], [73, 200], [70, 190], [74, 190],
[73, 200], [73, 200], [73, 184], [75, 200], [75, 180], [74, 210], [76, 187], [73, 200], [74, 200], [75, 205]
```



```
[ ] print(len(baseball))
```

```
1015
```

```
[ ] # Câu 1: Tạo một 2D numpy array tên np_baseball từ baseball.  
np_baseball=np.array(baseball)
```

```
# Xem kiểu dữ liệu (type) của np_baseball
```

```
print('Type:', type(np_baseball))
```

```
# Xem kích thước (shape) của np_baseball
```

```
print('Shape: ', np_baseball.shape)
```

```
Type: <class 'numpy.ndarray'>
```

```
Shape: (1015, 2)
```

► Nhấn vào đây để xem kết quả !

```
[ ] # Câu 2: In các giá trị của dòng thứ 50 trong np_baseball.  
print(np_baseball[49])
```

```
[ 70 195]
```

```
[ ] # Câu 3: Tạo một numpy array np_weight với dữ liệu được lấy từ cột hai của np_baseball.  
np_weight=np.array(np_baseball[:,1])  
# In danh sách các phần tử của np_weight.  
print(np_weight)
```

```
[180 215 210 ... 205 190 195]
```

► Nhấn vào đây để xem kết quả !

```
[ ] # Câu 4: Cho biết chiều cao của vận động viên thứ 124, và in ra kết quả  
print(np_baseball[123,0])
```

```
[ ] # Câu 5: Cho biết chiều cao trung bình, cân nặng trung bình của các cầu thủ
print('Cân nặng trung bình: ',np.mean(np_weight))
print('Chiều cao trung bình: ',np.mean(np_baseball[:,0]))
```

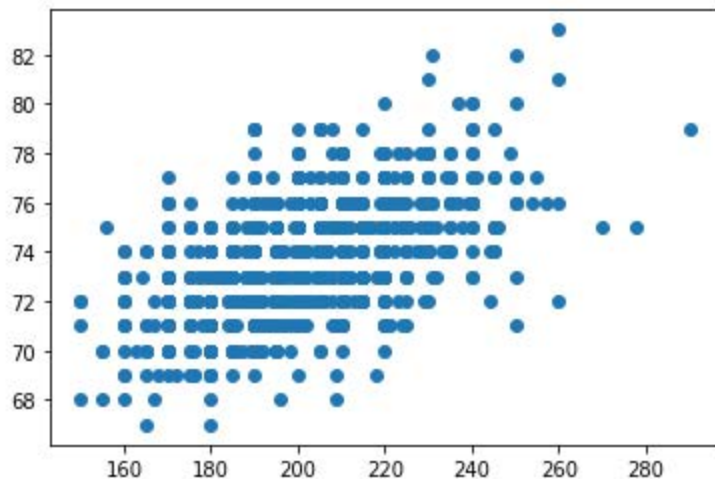
Cân nặng trung bình: 201.34876847290641
Chiều cao trung bình: 73.6896551724138

```
[ ] # Câu 6: Bạn nhận xét gì về mối tương quan giữa chiều cao và cân nặng của các cầu thủ:
# có/ không có tương quan, tương quan thuận/ngịch
```

- Cân nặng và chiều cao của cầu thủ có tương quan thuận.

#Plot

```
import matplotlib.pyplot as plt
plt.scatter(np_weight,np_baseball[:,0])
plt.show()
```



Chapter 3 - Exercise 5: Tính median của chiều cao (height) dựa vào vị trí (position)

Các kiến thức sử dụng trong bài tập:

Các xử lý trên mảng

1. Lọc các giá trị của mảng theo điều kiện
2. Tính toán thông kê trên mảng

Cho 2 tập tin heights.txt và positions.txt => chép dữ liệu từ 2 tập tin vào 2 list là heights và positions, sau đó thực hiện các yêu cầu, và đối chiếu với kết quả được cung cấp:

'GK' (goalkeeper), 'M' (midfield), 'A' (attack) and 'D' (defense).

```
[ ] import numpy as np
heights = [191, 184, 185, 180, 181, 187, 170, 179, 183, 186, 185, 170, 187, 183, 173, 188, 183, 180, 188, 175,
193, 180, 185, 170, 183, 173, 185, 185, 168, 190, 178, 185, 185, 193, 183, 184, 178, 180, 177, 188, 177, 187,
186, 183, 189, 179, 196, 190, 189, 188, 188, 188, 182, 185, 184, 178, 185, 193, 188, 179, 189, 188, 180, 178,
186, 188, 180, 185, 172, 179, 180, 174, 183, 178, 187, 178, 193, 181, 180, 187, 179, 173, 175, 188, 187, 175,
171, 179, 180, 188, 185, 196, 183, 184, 186, 178, 188, 168, 176, 178, 178, 192, 172, 170, 190, 175, 174, 179,
177, 187, 184, 185, 175, 193, 185, 191, 181, 183, 176, 176, 182, 192, 187, 170, 189, 171, 181, 183, 178, 182,
186, 191, 175, 179, 180, 181, 178, 193, 179, 181, 186, 190, 190, 192, 185, 178, 182, 171, 182, 173, 192, 175,
183, 183, 184, 176, 183, 186, 178, 185, 188, 193, 193, 170, 188, 196, 175, 180, 184, 173, 180, 190, 186, 182,
183, 195, 188, 187, 190, 180, 194, 182, 182, 183, 178, 183, 171, 185, 177, 180, 195, 173, 185, 186, 187, 178,
185, 174, 175, 176, 191, 170, 183, 180, 174, 191, 179, 178, 187, 191, 183, 180, 184, 183, 180, 185, 184, 181,
186, 185, 182, 175, 173, 175, 176, 174, 184, 177, 185, 162, 180, 171, 183, 180, 180, 191, 196, 191, 176, 186,
171, 190, 188, 180, 185, 176, 187, 188, 182, 178, 176, 175, 177, 191, 183, 189, 173, 180, 180, 185, 185, 180,
181, 183, 180, 185, 175, 175, 177, 177, 182, 167, 176, 180, 194, 180, 187, 174, 182, 174, 181, 188, 188, 180]
```



```
[ ] positions = ['GK', 'M', 'A', 'D', 'M', 'D', 'M', 'M', 'M', 'A', 'M', 'M', 'A', 'A', 'A', 'M', 'D', 'A', 'D', 'M',  
                'GK', 'D', 'D', 'M', 'M', 'M', 'M', 'D', 'M', 'GK', 'D', 'GK', 'D', 'D', 'M', 'A', 'M', 'D', 'M', 'GK', 'M',  
                'GK', 'A', 'D', 'GK', 'A', 'GK', 'GK', 'GK', 'GK', 'A', 'D', 'A', 'D', 'D', 'M', 'D', 'M', 'D', 'D', 'GK', 'GK',  
                'D', 'M', 'M', 'GK', 'M', 'D', 'M', 'M', 'D', 'D', 'M', 'M', 'D', 'A', 'A', 'M', 'M', 'M', 'A', 'D', 'D', 'A',  
                'A', 'M', 'M', 'M', 'D', 'D', 'A', 'A', 'D', 'M', 'M', 'M', 'D', 'M', 'M', 'D', 'M', 'A', 'M', 'M', 'GK', 'M',  
                'D', 'M', 'M', 'D', 'M', 'M', 'A', 'GK', 'D', 'M', 'GK', 'M', 'M', 'M', 'M', 'D', 'D', 'M', 'D', 'M', 'D', 'M',  
                'M', 'A', 'M', 'GK', 'A', 'M', 'D', 'M', 'D', 'GK', 'D', 'D', 'M', 'A', 'GK', 'M', 'D', 'A', 'D', 'A', 'A', 'M',  
                'D', 'M', 'A', 'GK', 'D', 'M', 'GK', 'A', 'D', 'D', 'D', 'GK', 'GK', 'M', 'D', 'GK', 'D', 'M', 'GK', 'A', 'D',  
                'GK', 'GK', 'D', 'M', 'GK', 'D', 'D', 'D', 'M', 'D', 'M', 'D', 'D', 'A', 'D', 'D', 'D', 'M', 'M', 'A', 'D', 'M',  
                'M', 'D', 'M', 'A', 'A', 'D', 'A', 'GK', 'M', 'A', 'A', 'D', 'D', 'A', 'D', 'GK', 'D', 'M', 'D', 'D', 'M', 'M',  
                'GK', 'D', 'M', 'GK', 'GK', 'D', 'M', 'D', 'D', 'M', 'A', 'D', 'D', 'M', 'A', 'A', 'A', 'A', 'A', 'M', 'D', 'D',  
                'A', 'M', 'GK', 'M', 'GK', 'A', 'A', 'GK', 'M', 'D', 'M', 'D', 'D', 'M', 'M', 'A', 'A', 'D', 'D', 'D', 'M', 'M',  
                'GK', 'D', 'M', 'M', 'D', 'D', 'D', 'M', 'M', 'M', 'D', 'M', 'A', 'A', 'D', 'D', 'M', 'GK', 'A', 'D', 'D', 'D',  
                'GK', 'D', 'M', 'D', 'A', 'A', 'GK', 'A', 'D', 'M', 'M', 'GK', 'A', 'A', 'M', 'D', 'A', 'M', 'M', 'M', 'D', 'D',  
                'D', 'M', 'D', 'A', 'M', 'M', 'M', 'A', 'M', 'M', 'D', 'M', 'D', 'M', 'M', 'A', 'D', 'D', 'M', 'A', 'D', 'D',  
                'M', 'M', 'M', 'D', 'M', 'D', 'A', 'D', 'D', 'M', 'D', 'A', 'D', 'D', 'GK', 'M', 'M', 'M', 'GK', 'M', 'A', 'D',  
                'D', 'M', 'A', 'GK', 'M', 'D', 'A', 'M', 'A', 'A', 'A', 'M', 'GK', 'A', 'A', 'M', 'A', 'D', 'D', 'D', 'A', 'GK',
```

```
[ ] # Câu 1:  
# a) Tạo numpy array np_positions từ list positions  
np_positions=np.array(positions,dtype='U5')  
  
# In danh sách các phần tử của np_positions  
print(np_positions)  
# Xem kiểu dữ liệu (type) của np_positions  
print(type(np_positions))  
  
print('')  
  
# b) Tạo numpy array np_heights từ list heights  
np_heights=np.array(heights,dtype='float')  
  
# In danh sách các phần tử của np_heights  
print(np_heights)  
  
# Xem kiểu dữ liệu (type) của np_heights  
print(type(np_heights))
```

```
['GK' 'M' 'A' ... 'D' 'D' 'M']
```

```
<class 'numpy.ndarray'>
```

```
[191. 184. 185. ... 183. 179. 179.]
```

```
<class 'numpy.ndarray'>
```

```
[ ] # Câu 2: Tính chiều cao trung bình của các GK (Goal Keeper).  
GK_heights=np_heights[np_positions=='GK']  
np.median(GK_heights)
```

188.0

```
[ ] # Câu 3: Tính chiều cao trung bình của những vị trí khác (Không phải là Goal Keeper).  
other_heights=np_heights[np_positions!='GK']  
np.median(other_heights)
```

181.0

```
[ ] # Câu 4: Tạo mảng dữ liệu có cấu trúc tự định nghĩa players gồm 'position' kiểu văn bản (U5) và 'height' kiểu 'float'  
  
dt=np.dtype([('position','U5'),('height','float')])  
arr_players=np.empty([len(np_positions)],dtype=dt)  
  
arr_players['position']=np_positions  
arr_players['height']=np_heights  
  
print(arr_players)
```

```
[('GK', 191.) ('M', 184.) ('A', 185.) ... ('D', 183.) ('D', 179.)  
 ('M', 179.)]
```

```
[ ] # Câu 5: Sắp mảng players theo height, cho biết vị trí có chiều cao cao nhất và chiều cao thấp nhất
print('Mảng players theo chiều cao: ', np.sort(arr_players, order='height'))
print('Chiều cao cao nhất:', max(arr_players['height']))
print('Chiều cao thấp nhất: ', min(arr_players['height']))
```

```
Mảng players theo chiều cao: [('M', 158.) ('A', 160.) ('M', 160.) ... ('A', 203.) ('GK', 203.)
('GK', 208.)]
Chiều cao cao nhất: 208.0
Chiều cao thấp nhất: 158.0
```



```
[ ] import pandas as pd
df = pd.DataFrame(arr_players)
df.head()
```

	position	height
0	GK	191.0
1	M	184.0
2	A	185.0
3	D	180.0
4	M	181.0

```
[ ] df['height'].min()
```

158.0

```
[ ] df.loc[df['height']==df['height'].min()]
```

	position	height
3367	M	158.0

```
[ ] a = np.array(np_positions, dtype='<U5')
    print (a)
    b = np.array(np_heights, dtype=float)
    print (b)
    players = np.vstack((a,b)).T
    players[0]
```

```
['GK' 'M' 'A' ... 'D' 'D' 'M']
[191. 184. 185. ... 183. 179. 179.]
array(['GK', '191.0'], dtype='<U32')
```