


▼ Chapter 2 - Exercise 1:

Tạo danh sách 5 website mà bạn thường xuyên truy cập kèm theo link của chúng

▼ cú pháp để tạo liên kết trong Markdown

[nội dung mô tả](địa chỉ trang web)

Sử dụng dữ liệu được cung cấp sau đây để tạo liên kết đến 5 website tương ứng:



1. [Udemy.com](https://www.udemy.com/)	1. Udemy.com
2. [BigData University](https://cognitiveclass.ai/)	2. BigData University
3. tuoitre.vn	3. tuoitre.vn
4. [google.com](https://www.google.com/)	4. google.com
5. vnexpress.net	5. vnexpress.net

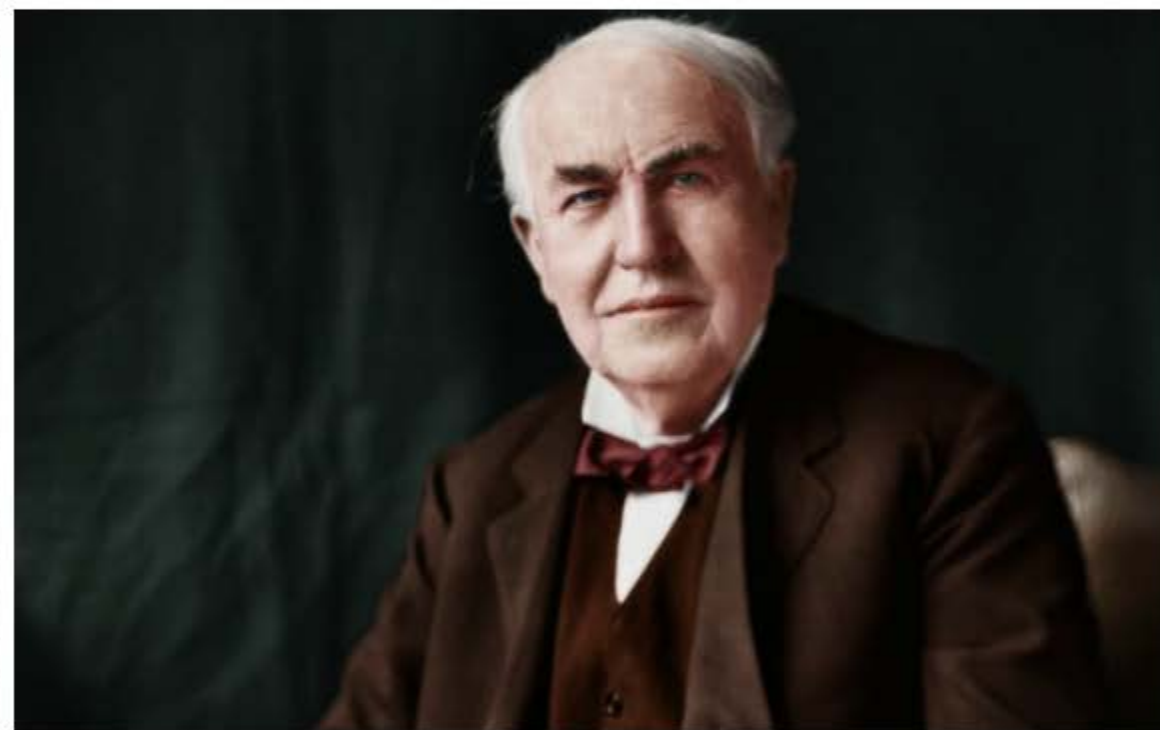
+ Code + Text

Connect Editing ^

Sử dụng dữ liệu được cung cấp sau đây để mô tả tiểu sử của nhà khoa học Thomas Alva Edison:

```
## Thomas Alva Edison (February 11, 1847 – October 18, 1931)
![[Thomas Edison](https://cdn-images-1.medium.com/max/1000/1*s2GyMoLeSV3Epc2Gk3qBXA.png)]
[Visit Thomas Edison Wikipedia](https://en.wikipedia.org/wiki/Thomas_Edison)
* Born: Thomas Alva Edison, February 11, 1847, Milan, Ohio, U.S.
* Died: October 18, 1931 (aged 84), West Orange, New Jersey, U.S.
* Burial place: Thomas Edison National Historical Park, Nationality American
* Education: Self-educated
* Occupation: Inventor, businessman
```

Thomas Alva Edison (February 11, 1847 – October 18, 1931)



[Visit Thomas Edison Wikipedia](https://en.wikipedia.org/wiki/Thomas_Edison)

- Born: Thomas Alva Edison, February 11, 1847, Milan, Ohio, U.S.
- Died: October 18, 1931 (aged 84), West Orange, New Jersey, U.S.
- Burial place: Thomas Edison National Historical Park, Nationality American
- Education: Self-educated
- Occupation: Inventor, businessman

+ Code + Text

RAM Disk Editing

SEARCH STACK OVERFLOW

► Nhấn vào đây để xem kết quả!

⌨
B
I
<>
🔗
🖼
☰
☰
☰
⋮
ψ
😊
💬

↑
↓
🔗
💬
⚙
✂
📄
🗑
⋮

```
STT|Tên khóa học|Thời lượng|
:---|:---:|---:|
1 | Fundamentals of Python | 36 |
2 | Python for Machine Learning, Data Science & Data Visualization| 36 |
3 | Maths & Statistic for Data Science | 32 |
4 | Databases & SQL for Data Science | 32 |
5 | Data Preprocessing & Analysis | 40 |
6 | Machine Learning with Python | 48 |
7 | R Programming Language for Data Science | 48 |
8 | Big Data in Machine Learning | 36 |
9 | Capstone Project | 50 |
```

STT	Tên khóa học	Thời lượng
1	Fundamentals of Python	36
2	Python for Machine Learning, Data Science & Data Visualization	36
3	Maths & Statistic for Data Science	32
4	Databases & SQL for Data Science	32
5	Data Preprocessing & Analysis	40
6	Machine Learning with Python	48
7	R Programming Language for Data Science	48
8	Big Data in Machine Learning	36
9	Capstone Project	50

Linear Regression

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$

$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

Decision Tree

- Entropy using frequency table of one attribute:

$$E(S) = -\sum_{i=1}^C p_i \log_2 p_i$$

- Entropy using frequency table of two attribute:

$$E(T, X) = -\sum_{c \in X} P(c) \log_2 P(c)$$

- Gain

$$\text{Gain}(T, X) = \text{Entropy}(T) - \text{Entropy}(E, X)$$

Linear Regression

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$

$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

Decision Tree

- Entropy using frequency table of one attribute:

$$E(S) = -\sum_{i=1}^C p_i \log_2 p_i$$

- Entropy using frequency table of two attribute:

$$E(T, X) = -\sum_{c \in X} P(c) \log_2 P(c)$$

- Gain

$$\text{Gain}(T, X) = \text{Entropy}(T) - \text{Entropy}(E, X)$$

▼ Chapter 3 - Exercise 1a: 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.array([0,1,2,3,4,5,6,7,8,9])  
# Hiển thị các phần tử có trong arr  
print("Cac phan tu cua array:",arr)  
# Xem kiểu dữ liệu (type) của arr  
print(type(arr))  
# Xem kích thước (shape) của arr  
print("Kich thuoc shape: ",arr.shape)
```

```
Cac phan tu cua array: [0 1 2 3 4 5 6 7 8 9]  
<class 'numpy.ndarray'>  
Kich thuoc shape: (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("Ptu trong arr_odd:\t",arr_odd)
print("Ptu trong arr_even:\t",arr_even)
```

```
Ptu trong arr_odd:      [1 3 5 7 9]
Ptu trong arr_even:     [0 2 4 6 8]
```

► Nhấn vào đây để xem kết quả!

```
[ ] # 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 = arr.copy()
arr_update[arr_update%2==1]=100

# Hiển thị các phần tử có trong arr_update_1
print("arr_updated", arr_update)
```

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

Chapter 3 - Exercise 1b: 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]
arr_a = [1,2,3,2,3,4,3,4,5,6]
arr_b = [7,2,10,2,7,4,9,4,9,8]

# Tạo arr_c chỉ lấy duy nhất các phần tử xuất hiện ở cả array arr_a và array arr_b
arr_c = np.intersect1d(arr_a, arr_b)
print(arr_c)
```

[2 4]

► Nhấn vào đây để xem kết quả!

```
[ ] # 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]


```
[ ] # Câu 3: Cho array arr_e = np.array([2, 6, 1, 9, 10, 3, 27, 8, 6, 25, 16])  
arr_e = np.array([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_f = arr_e[arr_e>=5]  
print("arr_f:",arr_f)
```

```
arr_f: [ 6  9 10 27  8  6 25 16]
```

Chapter 3 - Exercise 1c: 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,),dtype=np.int)
print ("arr_zeros:",arr_zeros)
```

```
arr_zeros: [0 0 0 0 0 0 0 0 0 0]
```

C:\Users\Abc\AppData\Local\Temp\ipykernel_4804\2743076336.py:2: DeprecationWarning: `np.int` is a deprecated alias for the builtin `int`. To silence this warning, use `int` by itself. Doing this will not modify any behavior and is safe. When replacing `np.int` with `int` you may wish to use `isinstance(x, int)` to check for `int` on older versions of Python (`np.int` still exists for backwards compatibility).

Deprecated in NumPy 1.20; for more details and guidance: <https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations>

```
arr_zeros = np.zeros((10,),dtype=np.int)
```

▶ 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(10,24)
print("arr_h:",arr_h)
```

In danh sách các phần tử theo tứ tự đảo ngược của `arr_h` vừa tạo

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

▶ `# Câu 3: Cho array arr_k = np.array([1, 2, 0, 8, 2, 0, 1, 3, 0, 5, 0])`
`arr_k = np.array([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_l = arr_k[arr_k!=0]`
`print("Arr_l:",arr_l)`

▶ Arr_l: [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("new arr_l:",arr_l)
```

new arr_l: [1 2 8 2 1 3 5 10 20]

▶ Nhấn vào đây để xem kết quả!

```
[ ] # Câu 5: Từ array của câu 4, thêm phần tử có giá trị 100 vào vị trí có index = 5
arr_l = np.insert(arr_l,5,100)
print(arr_l)
```

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

```
[ ] # Câu 6: Từ array của câu 5, xóa các phần tử tại vị trí có index = 0, 1, 2
lstIndex = [0,1,3]
arr_1 = np.delete(arr_1,lstIndex)
print(arr_1)
```

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



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

Chapter 3 - Exercise 2: Đọc và chuyển dữ liệu, sau đó tính BMI theo điều kiện, truy xuất dữ liệu, và đối chiếu với kết quả cho trước

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* => hãy chép dữ liệu từ tập tin này vào list là *height*, *weight*

```
# 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,
```

```
# 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,
```

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

```
1015
```

```
1015
```

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

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

```
arr_height: [74 74 72 ... 75 75 73]
kích thước shape của arr_height: (1015,)
```

► Nhấn vào đây để xem kết quả!

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

```
arr_weight: [180 215 210 ... 205 190 195]
kích thước shape của arr_weight: (1015,)
```

► Nhấn vào đây để xem kết quả!

```
[ ] # 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 * 0.0254
# 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]
```

```
[ ] # 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("Gia tri BMI:",arr_bmi)
```

```
Gia tri BMI: [23.11037639 27.60406069 28.48080465 ... 25.62295933 23.74810865
25.72686361]
```

► Nhấn vào đây để xem kết quả!

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

```
Gia tri o vi tri 50: 90.7184
```




```
# 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) từ  
arr_height_m_100 = arr_height_m[np.arange(100,111)]  
print("cac ptu tu 100 den 110 lay ca index 110",arr_height_m_100)
```



```
cac ptu tu 100 den 110 lay ca index 110 [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  
print("cau thu co BMI < 21:",arr_bmi[arr_bmi<21])  
# Áp dụng biểu thức điều kiện đã tạo để in ra các cầu thủ bóng chày có bmi < 21 trong arr_bmi
```

```
cau thu co BMI < 21: [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`
`print("Chiều cao trung bình:", np.mean(arr_height_m))`
`print("Cân nặng trung bình:", np.mean(arr_weight_kg))`

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

▶ Nhấn vào đây để xem kết quả!

[] `# Câu 10: Cho biết chiều cao và cân nặng lớn nhất của các cầu thủ`
`print("Chiều cao lớn nhất: %.2f" % np.amax(arr_height_m))`
`print("Cân nặng lớn nhất: %.2f" % np.amax(arr_weight_kg))`

Chiều cao lớn nhất: 2.11
Cân nặng lớn nhất: 131.54

▶ Nhấn vào đây để xem kết quả!

[] `# Câu 11: Cho biết chiều cao và cân nặng nhỏ nhất của các cầu thủ`
`print("Chiều cao nhỏ nhất: %.2f" % np.amin(arr_height_m))`
`print("Cân nặng nhỏ nhất: %.2f" % np.amin(arr_weight_kg))`

Chiều cao nhỏ nhất: 1.70
Cân nặng nhỏ nhất: 68.04

Chapter 3 - Exercise 3: Thao tác trên mảng nhiều chiều, 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.  
arr = np.ones([3,3],dtype=bool)  
# Hiển thị các phần tử của array arr  
print("arr 3x3 giá trị True:\n",arr)
```

```
arr 3x3 giá trị True:  
[[ 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.array([0, 1, 2, 3, 4, 5, 6, 7, 8])
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("Arr_2D:")
print(arr_2D)
# Trong arr_2D. Chuyển cột 1 sang cột 3 và ngược lại. => Xem lại danh sách các phần tử của arr_2D
arr_2D = arr_2D[:,[2,1,0]]
print("Arr2D sau chuyen doi cot:")
print(arr_2D)
```

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

Arr_2D:

[[0 1 2]

[3 4 5]

[6 7 8]]

Arr2D sau chuyen doi cot:

[[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 = arr_2D[[1,0,2],:]
# Xem lại danh sách các phần tử của arr_2D
print("Ds Arr_2D sau chuyen dong:")
print(arr_2D)
```

Ds Arr_2D sau chuyen dong:

```
[[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 = arr_2D[[2,1,0],:]
# Xem lại danh sách các phần tử của arr_2D
print("Ds arr_2D:")
print(arr_2D)
```

Ds 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
arr_2D = arr_2D[:,[2,1,0]]
print("Ds dao nguoc cac cot:")
print(arr_2D)
```

Ds dao nguoc cac cot:

```
[[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]]),
arr_2D_null = np.array([[1, 2, 3], [np.NaN, 5, 6], [7, np.NaN, 9], [4, 5, 6]])
print(arr_2D_null)
# Kiểm tra xem trong array có giá trị rỗng nào không?
ck = np.isnan(arr_2D_null).any()
if (ck == True):
    print("Array co gia tri rong")
else:
    print("Array ko co gia tri rong")
```

```
[[ 1.  2.  3.]
 [nan  5.  6.]
 [ 7. nan  9.]
 [ 4.  5.  6.]]
Array co gia tri rong
```



```
# 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 <
print("Array chưa null")
print(arr_2D_null)
arr_2D_null[np.isnan(arr_2D_null)]=0
print("Array thay the 0:")
print(arr_2D_null)
```



```
Array chưa null
[[1. 2. 3.]
 [0. 5. 6.]
 [7. 0. 9.]
 [4. 5. 6.]]
Array thay the 0:
[[1. 2. 3.]
 [0. 5. 6.]
 [7. 0. 9.]
 [4. 5. 6.]]
```

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

Cho tập tin *baseball_2D.txt* => chép dữ liệu từ tập tin vào list là *baseball*


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

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

```
[ ] 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],
```


 `print(len(baseball))`

 1015

+ Code

+ Text

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

 `<class 'numpy.ndarray'>
(1015, 2)`


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

```
[ 70 195]
```

► Nhấn vào đây để xem kết quả!

```
[ ] # Câu 3: Tạo một numpy array np_weight với dữ liệu được lấy từ cột hai của np_baseball.  
x,y = zip(*np_baseball)  
np_height = np.array(x)  
np_weight = np.array(y)  
# In danh sách các phần tử của np_weight.  
print("Ds ptu np_weight:")  
print(np_weight)
```

```
Ds ptu 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("Chiều cao của van dong vien thu 124: %d inch"%(x[123]))
```

```
Chiều cao của van dong vien thu 124: 75 inch
```




```
# Câu 5: Cho biết chiều cao trung bình, cân nặng trung bình của các cầu thủ
print("chiều cao trung bình:", np.mean(np_height))
print("cân nặng trung bình:", np.mean(np_weight))
```

```
chiều cao trung bình: 73.6896551724138
cân nặng trung bình: 201.34876847290641
```

► Nhấn vào đây để xem kết quả!

```
[ ] # 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/nghịch
a = np.cov(np_baseball)
a
```

```
array([[5618. , 7473. , 7314. , ..., 6890. , 6095. , 6466. ],
       [7473. , 9940.5, 9729. , ..., 9165. , 8107.5, 8601. ],
       [7314. , 9729. , 9522. , ..., 8970. , 7935. , 8418. ],
       ...,
       [6890. , 9165. , 8970. , ..., 8450. , 7475. , 7930. ],
       [6095. , 8107.5, 7935. , ..., 7475. , 6612.5, 7015. ],
       [6466. , 8601. , 8418. , ..., 7930. , 7015. , 7442. ]])
```



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

```
[ ] # Câu 1:
# a) Tạo numpy array np_positions từ list positions
np_positions = np.array(positions)
# In danh sách các phần tử của np_positions
print("Danh sach ptu cua np_position")
print(np_positions)
# Xem kiểu dữ liệu (type) của np_positions
print(type(np_positions))

# b) Tạo numpy array np_heights từ list heights
np_heights = np.array(heights)
# In danh sách các phần tử của np_heights
print("Danh sach ptu cua np_heights")
print(np_heights)
# Xem kiểu dữ liệu (type) của np_heights
print(type(np_heights))
```

```
Danh sach ptu cua np_position
['GK' 'M' 'A' ... 'D' 'D' 'M']
<class 'numpy.ndarray'>
Danh sach ptu cua np_heights
[191 184 185 ... 183 179 179]
<class 'numpy.ndarray'>
```

```
[ ] # Câu 2: Tính chiều cao trung bình của các GK (Goal Keeper).
a = np_heights[np_positions=="GK"].mean()
print("chiều cao trung bình của GK:",a)
```

chiều cao trung bình của GK: 188.23333333333332

► Nhấn vào đây để xem kết quả!

```
[ ] # Câu 3: Tính chiều cao trung bình của những vị trí khác (Không phải là Goal Keeper).
b = np_heights[np_positions!="GK"].mean()
print("chiều cao trung bình của GK:",b)
```

chiều cao trung bình của GK: 180.98888467853985

► Nhấn vào đây để xem kết quả!

```
[ ] # Câu 4: Tạo mảng dữ liệu có cấu trúc tự định nghĩa players gồm 'position' kiểu văn bản (U5) và 'height' kiểu float

# dtPos = np.dtype([('position','U5')])
# Ds = np.array(np_positions,dtype=dtPos)
# print(Ds)

dtPos = np.dtype([('position','U5'),('height',float)])
arr = list(zip(positions,heights))
Ds = np.array(arr,dtype=dtPos)
print(Ds)
```

```
[('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("Sap mang theo height:")
print(np.sort(Ds,order='height'))
print('Vi tri co chieu cao nhat:',np_positions[np_heights==max(np_heights)])
print('Vi tri co chieu thap nhat:',np_positions[np_heights==min(np_heights)])
```



```
Sap mang theo height:
[('M', 158.) ('A', 160.) ('M', 160.) ... ('A', 203.) ('GK', 203.)
 ('GK', 208.)]
Vi tri co chieu cao nhat: ['GK']
Vi tri co chieu thap nhat: ['M']
```