

Lliurament tasca 2-A - Estructura d'una Matriu

April 19, 2021

1 Lliurament tasca 2-A: Estructura d'una Matriu

1.1 NIVELL 1

Treballem els conceptes de l'estructura d'una matriu, dimensió, eixos i la vectorització que ens permet reduir l'ús de for loops en operacions aritmètiques o matemàtiques.

EXERCICI 1:

Crea un np.array d'una dimensió, que inclogui l'almenys 8 nombres sencers, data type int64. Mostra la dimensió i la forma de la matriu.

```
[13]: import numpy as np

array = np.random.randint(100,size = 18)
array_int64= np.int64(array)
print(f'El vector creat és el següent:\n{array_int64} y el tipus de dada_
→{array_int64.dtype}.\n')

print(f'Dimensió de la matriu: {array_int64.ndim}')
print(f'Forma de la matriu: {array_int64.shape}. És a dir, {array_int64.
→shape[0]} columnes i 1 fila. ')
```

El vector creat és el següent:

[9 20 75 6 83 99 0 60 36 68 52 78 6 35 29 70 23 83] y el tipus de dada int64.

Dimensió de la matriu: 1

Forma de la matriu: (18,). És a dir, 18 columnes i 1 fila.

EXERCICI 2:

De la matriu de l'exercici 1, calcula el valor mitjà dels valors introduïts i resta la mitjana resultant de cada un dels valors de la matriu.

```
[2]: print(f'Matriu creada:\n{array}\n')

mean_array = round(array.mean(),2)
print(f'Valor mitjà dels valors introduïts: {mean_array}')
```

```
print(f'\nResta la mitjana ({mean_array}) a cada un dels valors:  
→\n{array-mean_array}')
```

Matriu creada:

```
[93 29 80 41  3 66 12 87 48 89 85 39 23 10  7 41 27 87]
```

Valor mitjà dels valors introduïts: 48.17

Resta la mitjana (48.17) a cada un dels valors:

```
[ 44.83 -19.17  31.83  -7.17 -45.17  17.83 -36.17  38.83  -0.17  40.83  
 36.83  -9.17 -25.17 -38.17 -41.17  -7.17 -21.17  38.83]
```

EXERCICI 3:

Crea una matriu bidimensional amb una forma de 5 x 5. Extreu el valor màxim de la matriu, i els valors màxims de cadascun dels seus eixos.

```
[3]: matriu = np.random.randint(50,size = (5,5))  
print(f'Matriu creada (5 x 5):\n{matriu}\n')  
print(f'El valor màxim de la matriu és: {matriu.max()}')  
print(f'Els valors màxims de l'eix 0 (columnes) són: {matriu.max(0)}')  
print(f'Els valors màxims de l'eix 1 (files) són: {matriu.max(1)}')
```

Matriu creada (5 x 5):

```
[[41 22 24  1 22]  
 [ 1  4 41 14  3]  
 [43 13 48 27  7]  
 [21 32 46 11 22]  
 [45 28 42 35 18]]
```

El valor màxim de la matriu és: 48

Els valors màxims de l'eix 0 (columnes) són: [45 32 48 35 22]

Els valors màxims de l'eix 1 (files) són: [41 41 48 46 45]

1.2 NIVELL 2

Treballem els conceptes de l'estructura d'una matriu, Broadcasting, indexació, Mask..

EXERCICI 4:

Mostreu-me amb exemples de diferents matrius, la regla fonamental de Broadcasting que diu : “les matrius es poden transmetre / broadcast si les seves dimensions coincideixen o si una de les matrius té una mida d'1”.

El broadcasting és un procés automàtic implementat en les operacions de NumPy per eficientar els càlculs amb arrays multidimensionals. Es fa broadcasting encara que les matrius tinguin diferent forma, això sí, les seves dimensions han de coincidir o una de les matrius ha de tenir mida 1.

```
[4]: # Exemple 1: Broadcasting - Una de les matrius té mida 1.  
import numpy as np
```

```

a = np.arange(6).reshape(2,3)
print(f'Matriu a:\n{a}\n')

b = np.array([0,1,2])
print(f'Matriu b:\n{b}\n')

print(f'En aquest exemple, la matriu a i b tenen diferent dimensió.\nPerò, com
↳la la matriu b es de dimensió 1, Numpy fa ús de Broadcasting, com podem
↳veure a continuació:\n{a+b}'
)

```

Matriu a:

```

[[0 1 2]
 [3 4 5]]

```

Matriu b:

```

[0 1 2]

```

En aquest exemple, la matriu a i b tenen diferent dimensió.
Però, com la la matriu b es de dimensió 1, Numpy fa ús de Broadcasting, com
podem veure a continuació:

```

[[0 2 4]
 [3 5 7]]

```

[5]: *# Exemple 2: Broadcasting - Dues dimensions iguals*

```

a = np.array([0, 1, 2])
print(f'Matriu a:\n{a}\n')

b = np.array([5,5,5])
print(f'Matriu b:\n{b}\n')

print('En aquest exemple, la matriu a i b tenen la mateixa dimensió. Per tant,
↳Numpy fa ús de Broadcasting.')
print(a + b)

```

Matriu a:

```

[0 1 2]

```

Matriu b:

```

[5 5 5]

```

En aquest exemple, la matriu a i b tenen la mateixa dimensió. Per tant, Numpy fa
ús de Broadcasting.

```

[5 6 7]

```

EXERCICI 5:

Utilitza la Indexació per extreure els valors d'una columna i una fila de la matriu. I suma els seus valors.

```
[6]: print(f'Recordem la matriu de l'exercici 4:\n{matriu}')

print(f'\nSuma de la primera columna ({matriu[:,0]}): {matriu[:,0].sum()}')
print(f'Suma de la segona fila ({matriu[1]}): {matriu[1].sum()}')
```

Recordem la matriu de l'exercici 4:

```
[[41 22 24 1 22]
 [ 1  4 41 14  3]
 [43 13 48 27  7]
 [21 32 46 11 22]
 [45 28 42 35 18]]
```

Suma de la primera columna ([41 1 43 21 45]): 151

Suma de la segona fila ([1 4 41 14 3]): 63

EXERCICI 6

Mask la matriu anterior, realitzeu un càlcul booleà vectoritzat, agafant cada element i comprovant si es divideix uniformement per quatre.

Això retorna una matriu de mask de la mateixa forma amb els resultats elementals del càlcul.

```
[7]: mask = matriu %4 == 0
print(mask)
```

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

EXERCICI 7

A continuació, utilitzeu aquesta màscara per indexar a la matriu de números original. Això fa que la matriu perdi la seva forma original, reduint-la a una dimensió, però encara obteniu les dades que esteu cercant.

```
[8]: print(matriu[mask])
```

```
[24  4 48 32 28]
```

```
[9]: # Una altra forma de fer-ho més ràpid.
print(matriu[matriu % 4 == 0])
```

```
[24  4 48 32 28]
```

1.3 NIVELL 3

Manipulació d'imatges amb Matplotlib.

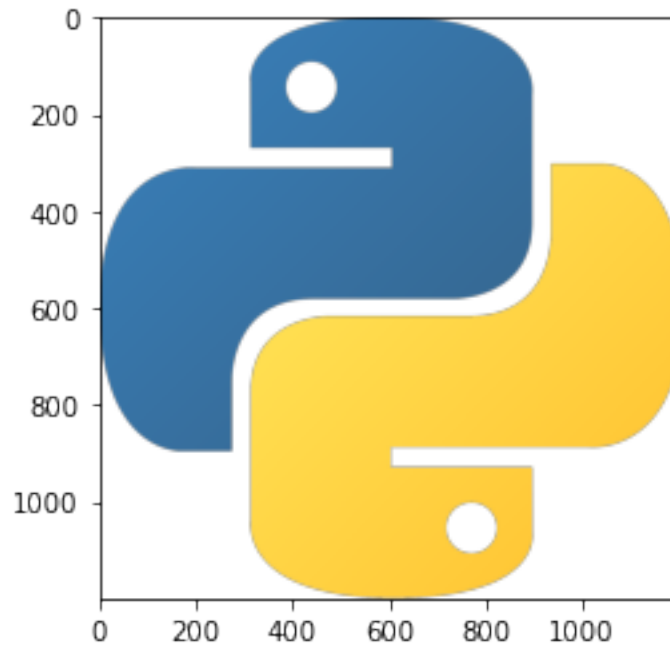
Carregareu qualsevol imatge (jpg, png ..) amb Matplotlib. adoneu-vos que les imatges RGB (Red, Green, Blue) són realment només amplades \times alçades \times 3 matrius (tres canals Vermell, Verd i Blau), una per cada color de nombres enters int8,

manipuleu aquests bytes i torneu a utilitzar Matplotlib per desar la imatge modificada un cop hàgiu acabat.

Ajuda:Importeu, `import matplotlib.image as mpimg`. estudieu el metotde `mpimg.imread()`

```
[10]: import matplotlib.pyplot as plt
import matplotlib.image as mpimg

img = mpimg.imread('python.png')
imgplot = plt.imshow(img)
plt.show()
```



EXERCICI 8

Mostreu-me a veure que passa quan eliminem el canal G Verd o B Blau.

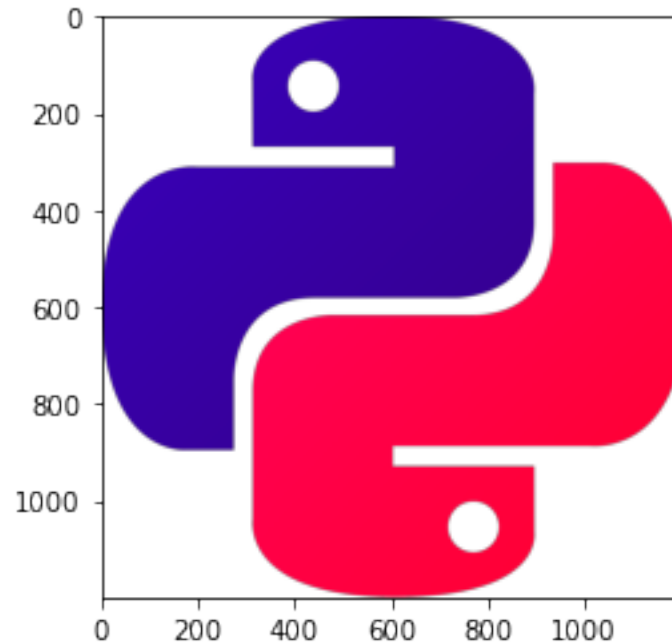
Mostreu-me a veure què passa quan eliminem el canal G Verd o B Blau. Hauries d'utilitzar la indexació per seleccionar el canal que voleu anul·lar.

Utilitzar el mètode, `mpimg.imsave()` de la llibreria importada, per guardar les imatges modificades i que haureu de pujar al vostre repositori a github.

```
[11]: # Elimino el color verd
img_exverd = np.copy(img)
```

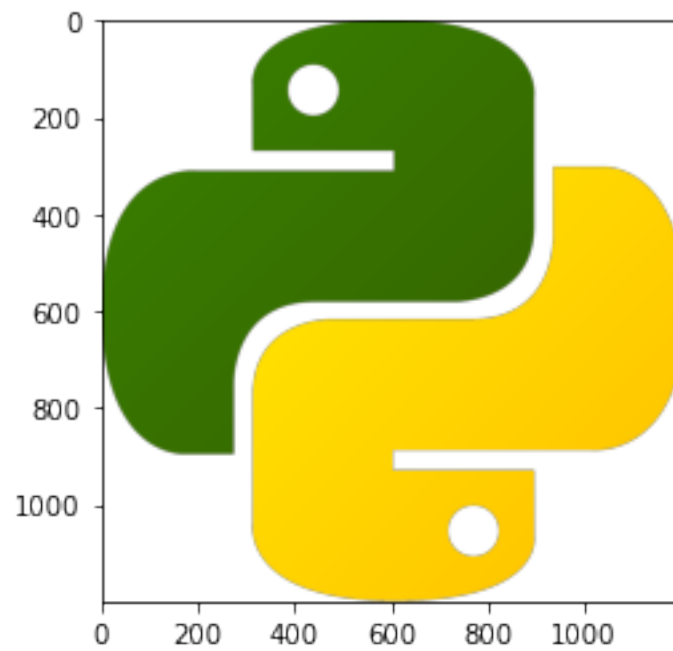
```
img_exverd[:, :, 1]=0

plt.show(plt.imshow(img_exverd))
mpimg.imsave('python1.jpg',img_exverd)
```



```
[12]: # Elimino el color blau
img_exblau = np.copy(img)
img_exblau[:, :, 2]=0
plt.show(plt.imshow(img_exblau))

mpimg.imsave('python2.jpg',img_exblau)
```



[]: