

# 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. .

In [5]:

```
import numpy as np

unaDimensio = np.array([3,8,95,4,8,78,2,15,6,98,2,36,54,78,95,25,29,32,30,22], dtype='int64')
print(unaDimensio)

print("La dimensió de la matriu és de", unaDimensio.ndim)

print("La forma de la matriu és de", unaDimensio.shape)
```

```
[ 3  8 95  4  8 78  2 15  6 98  2 36 54 78 95 25 29 32 30 22]
La dimensió de la matriu és de 1
La forma de la matriu és de (20,)
```

## - 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.

In [6]:

```
print("La mitjana de la matriu és", np.mean(unaDimensio))

print("Si restem la mitjana anterior a cadascun dels valors, la matriu queda així", unaDimensio - np.mean(unaDimensio))
```

```
La mitjana de la matriu és 36.0
Si restem la mitjana anterior a cadascun dels valors, la matriu queda així [-33. -28.  59. -32. -28.  42. -34. -21. -30.  62. -34.   0.  18.  42.  59. -11.  -7.  -4.  -6. -14.]
```

## - 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.

In [7]:

```
duesDimensions = np.random.randint(100, size=(2,5))
print(duesDimensions)

print("El valor màxim de la matriu és el", np.max(duesDimensions))

print("Els valors màxims de l'eix 0 són", np.max(duesDimensions, axis=0), " i els de l'eix 1 són", np.max(duesDimensions, axis=1))
```

```
[[72 97 65 53 33]
 [50 44 81 45 30]]
El valor màxim de la matriu és el 97
Els valors màxims de l'eix 0 són [72 97 81 53 33] i els de l'eix 1 són els [97 81]
```

# 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".

In [8]:

```
#Exemple 1: suma de matrius de tamany igual
matriuA = np.random.randint(10, size=(2,3))
matriuB = np.random.randint(5, size=(2,3))
print(matriuA)
print("_____")
print(matriuB)
print("_____")
novaMatriu = np.add(matriuA, matriuB)
print(novaMatriu)
```

```
[[2 3 2]
 [3 6 0]]
```

---

```
[[3 1 4]
 [2 0 0]]
```

---

```
[[5 4 6]
 [5 6 0]]
```

In [21]:

```
#Exemple 2: suma de matrius amb una de tamany amb mida 1
matriuA = np.random.randint(10, size=(2,3,4))
matriuB = np.random.randint(5, size=(2,1,1))
print(matriuA)
print("_____")
print(matriuB)
novaMatriu = np.add(matriuA, matriuB)
print("_____")
print(novaMatriu)
```

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

```
[[2 1 6 5]
 [0 1 9 6]
 [8 1 2 9]]]
```

---

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

```
[[3]]]
```

---

```
[[[ 6  7  3  8]
   [ 9  7  2  9]
   [ 7  3  2  4]]
```

```
[[ 5  4  9  8]
 [ 3  4 12  9]
 [11  4  5 12]]]
```

## - Exercici 5

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

In [9]:

```
matriuA = np.random.randint(10, size=(4,6))
matriuB = np.random.randint(10, size=(1,6))
print(matriuA)
print("_____")
print(matriuB)

novaMatriu = np.add(matriuA, matriuB)
print("_____")
print(novaMatriu)

#Extracció dels valors d'una columna
print("Els valors de la primera columna de la novaMatriu són", novaMatriu[0:5, 0])

#Suma dels valors d'una columna
sumaC = np.sum(novaMatriu[0:5,0])
print("La suma dels valors de la primera columna de la novaMatriu és de", sumaC)

#Extracció dels valors d'una fila
print("Els valors de la tercera fila de la novaMatriu són", novaMatriu[2:3])

#Suma dels valors d'una fila
sumaF = np.sum(novaMatriu[2:3])
print("La suma dels valors de la tercera fila de la novaMatriu és de", sumaF)

#Suma dels valors d'una columna i una fila
print("La suma dels valors de la primera columna i de la tercera fila de la novaMatriu és
```

```
[[6 1 4 6 3 9]
 [7 6 5 0 6 8]
 [9 3 6 2 7 2]
 [3 3 1 1 9 9]]
```

---

```
[[9 9 9 5 9 7]]
```

---

```
[[15 10 13 11 12 16]
 [16 15 14  5 15 15]
 [18 12 15  7 16  9]
 [12 12 10  6 18 16]]
```

Els valors de la primera columna de la novaMatriu són [15 16 18 12]

La suma dels valors de la primera columna de la novaMatriu és de 61

Els valors de la tercera fila de la novaMatriu són [[18 12 15 7 16 9]]

La suma dels valors de la tercera fila de la novaMatriu és de 77

La suma dels valors de la primera columna i de la tercera fila de la novaMatriu és de 138

## - 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.

In [10]:

```
print(novaMatriu)

print("_____")
mask = novaMatriu%4 ==0      #si el número dividit pel mòdul de 4 és 0, el número és divis
print(mask)
print("_____")
```

```
[[15 10 13 11 12 16]
 [16 15 14  5 15 15]
```

```
[18 12 15 7 16 9]
[12 12 10 6 18 16]]
```

---

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

---

## - 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.

In [11]:

```
#Extracció dels valors d'una columna
print("Els valors de la primera columna de la mask són", mask[0:5, 0])

print("Els números divisibles per 4 són:", novaMatriu[mask]) #imprimeix només els valors
```

```
Els valors de la primera columna de la mask són [False True False True]
Els números divisibles per 4 són: [12 16 16 12 16 12 12 16]
```

## 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 × alçades × 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. estudeu el metode mpimg.imread()

In [12]:

```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

tom = mpimg.imread(r'C:\Users\Anna\Desktop\DataScience\SPRINTS\SPRINT 2\SPRINT 2 T02\tom.-.jpg')

plt.imshow(tom) #imprimeix la imatge com una foto amb eixos x,y

print(tom) #imprimeix la imatge en forma de matriu

print("La mida de la imatge és", imatge.shape, ", els valors són de tipus", imatge.dtype,
      #imprimir aquestes característiques ens ajuda a modificar la imatge
```

```
[[[ 86  47  16]
   [ 88  49  18]
   [ 91  52  21]
   ...
   [251 234 218]
   [252 235 219]
   [254 235 220]]

[[[ 89  50  19]
   [ 90  51  20]
   [ 92  53  22]
   ...
   [252 235 219]
```

```

[254 237 221]
[255 236 221]]

[[ 94  55  24]
 [ 94  55  24]
 [ 95  56  25]
 ...
[255 237 223]
[255 238 224]
[255 238 224]]

...

[[186 157 113]
 [189 160 116]
 [193 164 120]
 ...
[226 201 160]
[227 202 161]
[227 202 161]]

[[187 160 115]
 [189 162 117]
 [194 167 122]
 ...
[228 203 162]
[228 203 162]
[228 203 162]]

[[187 160 115]
 [190 163 118]
 [194 167 122]
 ...
[229 204 163]
[229 204 163]
[230 205 164]]]

```

```

-----
NameError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_18400\3174619228.py in <module>
      8 print(tom)      #imprimeix la imatge en forma de matriu
      9
--> 10 print("La mida de la imatge és", imatge.shape, ", els valors són de tipus", imatge
.dtype, "i la imatge és de tipus", type(imatge))
     11 #imprimir aquestes característiques ens ajuda a modificar la imatge

```

NameError: name 'imatge' is not defined



In [13]:

```

#Fem zoom a la imatge
zoom = tom[1000:,:2000,:] #de l'eix X m'agafa des del pixel 1000, de l'eix Y des del pixel 2000

```

```
plt.imshow(zoom)
print(zoom)

plt.imsave('tomZoom.png',zoom) #guardem la imatge en format png
```

```
[[[245 247 244]
   [245 247 244]
   [245 247 244]
   ...
   [ 0 0 0]
   [ 0 0 0]
   [ 0 0 0]]

 [[245 247 244]
   [245 247 244]
   [246 248 245]
   ...
   [ 0 0 0]
   [ 0 0 0]
   [ 0 0 0]]

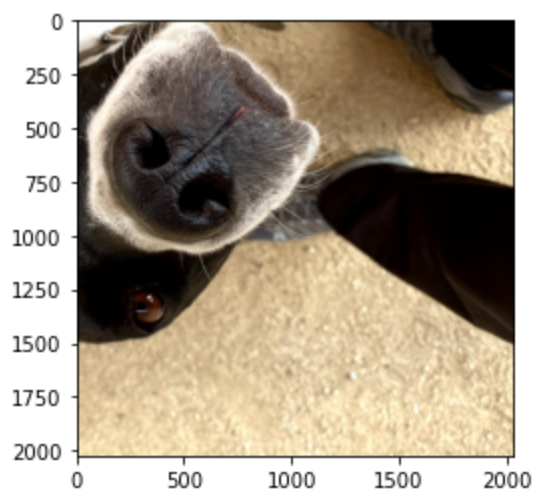
 [[246 248 245]
   [246 248 245]
   [246 248 245]
   ...
   [ 0 0 0]
   [ 0 0 0]
   [ 0 0 0]]

 ...

 [[254 250 202]
   [255 251 203]
   [255 252 204]
   ...
   [226 201 160]
   [227 202 161]
   [227 202 161]]

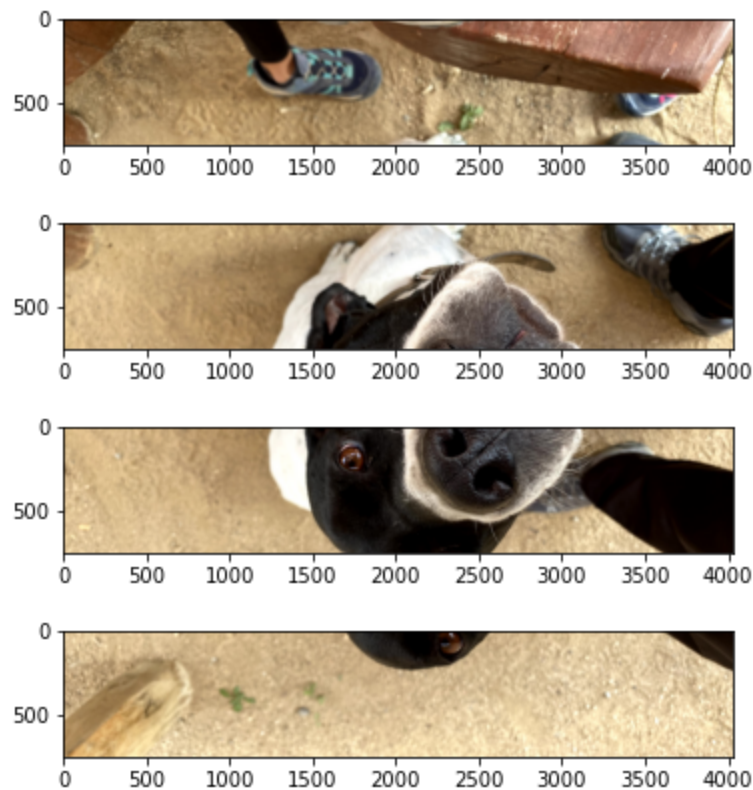
 [[255 252 204]
   [255 252 204]
   [255 253 205]
   ...
   [228 203 162]
   [228 203 162]
   [228 203 162]]

 [[255 252 204]
   [255 253 205]
   [255 254 206]
   ...
   [229 204 163]
   [229 204 163]
   [230 205 164]]]
```



In [21]:

```
#dividim la imatge amb 4 parts
p1,p2,p3,p4 = np.split(tom,4)
plt.imshow(p1)
plt.show() #hem d'afegir aquest codi perquè si no només s'imprimeix l'última imatge i vo.
plt.imshow(p2)
plt.show()
plt.imshow(p3)
plt.show()
plt.imshow(p4)
plt.show()
print(p1,p2,p3,p4)
plt.imsave('tomP1.png',p1)
plt.imsave('tomP2.png',p2)
plt.imsave('tomP3.png',p3)
plt.imsave('tomP4.png',p4)
```



```
[[ [ 86  47  16]
   [ 88  49  18]
   [ 91  52  21]
   ...
   [251 234 218]
   [252 235 219]
```

```

[254 235 220]]

[[ 89  50  19]
 [ 90  51  20]
 [ 92  53  22]
 ...
[252 235 219]
[254 237 221]
[255 236 221]]

[[ 94  55  24]
 [ 94  55  24]
 [ 95  56  25]
 ...
[255 237 223]
[255 238 224]
[255 238 224]]

...

[[139  96  62]
 [139  96  62]
 [138  95  61]
 ...
[250 231 188]
[249 231 185]
[249 231 185]]

[[139  96  62]
 [139  96  62]
 [138  95  61]
 ...
[251 230 187]
[248 230 184]
[247 229 183]]

[[139  96  62]
 [139  96  62]
 [138  95  61]
 ...
[249 228 185]
[248 227 182]
[248 227 182]] [[139  96  62]
 [139  96  62]
 [138  95  61]
 ...
[248 225 183]
[247 224 180]
[247 224 180]]

[[139  96  62]
 [139  96  62]
 [138  95  61]
 ...
[247 222 181]
[245 222 178]
[245 222 178]]

[[139  96  62]
 [139  96  62]
 [138  95  61]
 ...
[246 221 180]
[245 221 177]
[244 220 176]]

```



```

...
[[215 185 135]
 [215 185 135]
 [215 185 135]
...
[245 224 181]
[251 232 189]
[255 241 198]]

[[214 184 134]
 [214 184 134]
 [215 185 135]
...
[245 224 179]
[252 234 188]
[255 246 200]]

[[214 184 134]
 [214 184 134]
 [214 184 134]
...
[246 225 180]
[253 235 189]
[255 248 202]]] [[[213 183 133]
 [213 183 133]
 [214 184 134]
...
[239 218 173]
[250 232 186]
[255 240 194]]

[[213 183 133]
 [213 183 133]
 [214 184 134]
...
[238 217 172]
[248 230 184]
[255 238 192]]

[[213 183 133]
 [213 183 133]
 [214 184 134]
...
[235 214 167]
[244 226 178]
[251 233 185]]

...

[[221 194 149]
 [220 193 148]
 [218 191 146]
...
 [ 13    4    0]
 [ 13    4    0]
 [ 13    4    0]]

[[220 193 148]
 [219 192 147]
 [218 191 146]
...
 [ 13    4    0]
 [ 14    5    0]
 [ 14    5    0]]

```

```

[[219 192 147]
 [219 192 147]
 [218 191 146]
 ...
 [ 14   5   0]
 [ 14   5   0]
 [ 14   5   0]]] [[[218 191 146]
 [218 191 146]
 [218 191 146]
 ...
 [ 15   6   1]
 [ 15   6   1]
 [ 15   6   1]]

[[217 190 145]
 [218 191 146]
 [218 191 146]
 ...
 [ 16   7   2]
 [ 16   7   2]
 [ 16   7   2]]

[[217 190 145]
 [217 190 145]
 [218 191 146]
 ...
 [ 16   7   2]
 [ 16   7   2]
 [ 17   8   3]]

...

[[186 157 113]
 [189 160 116]
 [193 164 120]
 ...
 [226 201 160]
 [227 202 161]
 [227 202 161]]

[[187 160 115]
 [189 162 117]
 [194 167 122]
 ...
 [228 203 162]
 [228 203 162]
 [228 203 162]]

[[187 160 115]
 [190 163 118]
 [194 167 122]
 ...
 [229 204 163]
 [229 204 163]
 [230 205 164]]]

```

## - 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.

Objectius dimensions, formes Vectorització, Broadcasting Index i mask Manipulació imatges amb numpy import  
`cv2 image = cv2.imread("chelsea-the-cat.png")`

In [64]:

```
print(tom.shape)

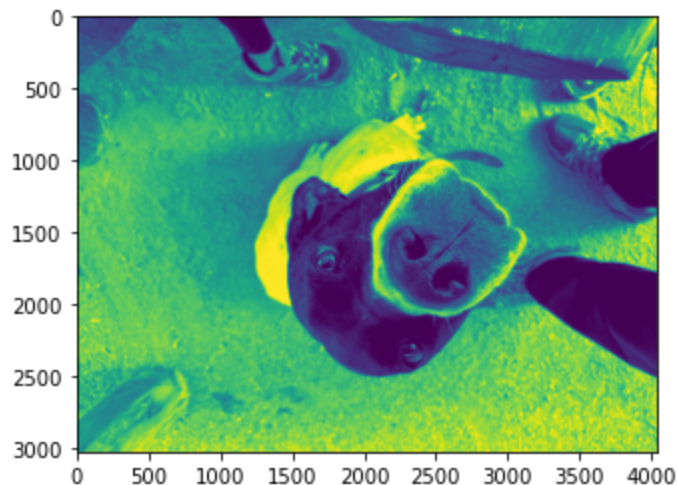
#ELIMINAR EL VERD

#NO EM DEIXA ELIMINAR NOMÉS EL VERD, NO ENTENC PERQUÈ, PERÒ SI QUE EM DEIXA NOMÉS IMPRIMIR EL VERD
# FAIG SERVIR EL SEGÜENT CODI tomG=tom[:, :, 0-2:] i em diu que el shape no és correcte per a tomG

tomG=tom[:, :, 1]

print(tomG)
plt.imshow(tomG)
plt.show()
plt.imsave('tomG.png', tomG)
```

```
(3024, 4032, 3)
[[ 47  49  52 ... 234 235 235]
 [ 50  51  53 ... 235 237 236]
 [ 55  55  56 ... 237 238 238]
 ...
 [157 160 164 ... 201 202 202]
 [160 162 167 ... 203 203 203]
 [160 163 167 ... 204 204 205]]
```



In [72]:

```
#ELIMINAR EL VERMELL

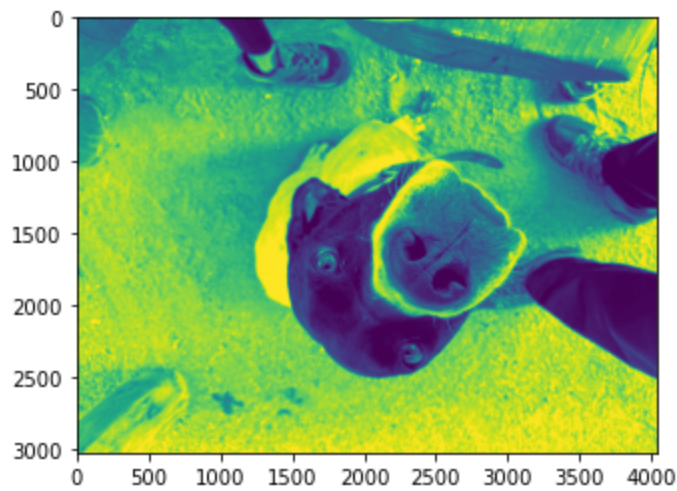
#NO EM DEIXA ELIMINAR NOMÉS EL VERMELL, NO ENTENC PERQUÈ, PERÒ SI QUE EM DEIXA NOMÉS IMPRIMIR EL VERMELL
# FAIG SERVIR EL SEGÜENT CODI tomR=tom[:, :, 1:] i em diu que el shape no és correcte per a tomR

tomR=tom[:, :, 0]

print(tomR)
plt.imshow(tomR)
plt.show()
plt.imsave('tomR.png', tomR)
```

```
[[ 86  88  91 ... 251 252 254]
 [ 89  90  92 ... 252 254 255]
 [ 94  94  95 ... 255 255 255]
 ...
 [186 189 193 ... 226 227 227]]
```

```
[187 189 194 ... 228 228 228]
[187 190 194 ... 229 229 230]]
```



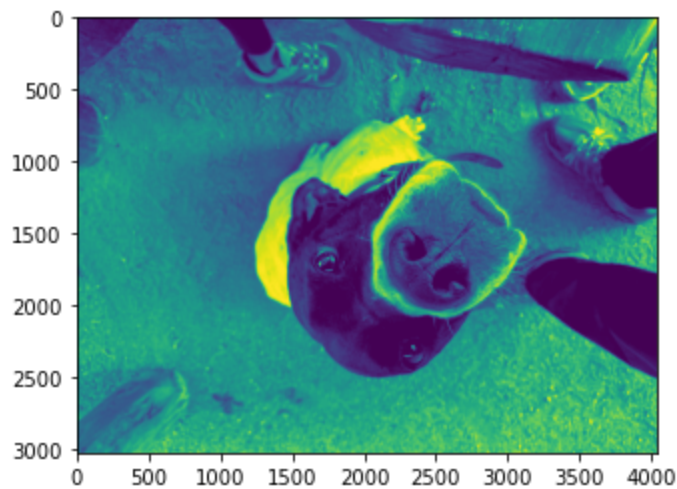
In [73]:

```
#ELIMINAR EL BLAU

#NO EM DEIXA ELIMINAR NOMÉS EL BLAU, NO ENTENC PERQUÈ, PERÒ SI QUE EM DEIXA NOMÉS IMPRIMI
# FAIG SERVIR EL SEGÜENT CODI tomB=tom[:, :, 0:2] i em diu que el shape no és correcte per
tomB=tom[:, :, 2]

print(tomB)
plt.imshow(tomB)
plt.show()
plt.imsave('tomB.png', tomB)
```

```
[ [ 16  18  21 ... 218 219 220]
  [ 19  20  22 ... 219 221 221]
  [ 24  24  25 ... 223 224 224]
  ...
  [113 116 120 ... 160 161 161]
  [115 117 122 ... 162 162 162]
  [115 118 122 ... 163 163 164]]
```



In [ ]: