In [7]:

```
#S03 T04: Pràctica amb programació numèrica
#Descripció
#Familiaritza't amb la Programació Numèrica a través de la llibreria NumPy.
```

In [8]:

```
#Nivell 1
```

In [28]:

```
#Exercici 1
#Crea una funció que donat un Array d'una dimensió, et faci un resum estadístic bàsic d
e les dades.
#Si detecta que l'array té més d'una dimensió, ha de mostrar un missatge d'error.

import numpy as np

def resum(v_array):
    if v_array.ndim > 1:
        print("La dimensió de l'array és més gran a 1")
    else:
        np.info(v_array)
        print("size:",np.size(v_array))
        print("type:",type(v_array.shape))

m_array = np.array([1,2,3,4,5])
resum(m_array)
3
```

```
class:  ndarray
shape:  (5,)
strides:  (4,)
itemsize:  4
aligned:  True
contiguous:  True
fortran:  True
data pointer: 0x283cf1e1e20
byteorder:  little
byteswap:  False
type: int32
size: 5
type: <class 'tuple'>
```

In [169]:

```
#Exercici 2
#Crea una funció que et generi un quadrat NxN de nombres aleatoris entre el 0 i el 100.

import numpy as np

def quadrat_NxN(t):

    if t > 0:
        quadrat = np.zeros(shape=(t,t)) # Cada element del quadrat s'inicialitza a 0.
        for i in range(t):
            for j in range(t):
                quadrat[i][j] = randint(0,100) # A cada element del quadrat l`hi assign
em un número aleatori.
    else:
        print("El tamany del quadrat no és correcte.")
        return
    print(quadrat)

tam = int(input("Escriu tamany del quadrat: "))
quadrat_NxN(tam)
```

```
Escriu tamany del quadrat: 10
[[ 22.   7.  89.  46.  45.  90.  54.  83.  76.  18.]
 [ 87.  38.  67.  22.  52.  29.  33.  54.  54.  36.]
 [ 19.  52.  46.  39.  13.   1.  26.  34.  82. 100.]
 [ 29.   2.  13.  50.  50.  80.  76.  48.  14.  15.]
 [ 81.  50.  32.  71.  89.  74.  75.  73.  27.  86.]
 [ 85.   7.  64.  37.  65.  51.  45.  20.  40.  15.]
 [ 27.  71.   1.  30.  95.  58.  99.  48.  27.  63.]
 [ 23.  86.  98.  74.  45.  12.  73.  68.  33.  62.]
 [ 64.   4.  84.  75.  11.  72.  74.   8.  49.  32.]
 [ 70.  93.  19.  53.  77.  70.  97.  60.  61.  61.]]
```

In [208]:

```
#Exercici 3
#Crea una funció que donada una taula de dues dimensions, et calculi els totals per fil
a i els totals per columna.

import numpy as np

def taula_2D(tau):

    tot_fila = 0
    tot_col = 0

    if(tau.ndim != 2):
        print("La taula no és bidimensional.")
    else:
        tot_fila = np.sum(tau, axis=1) # sumatori d'elements per files.
        tot_col = np.sum(tau, axis=0)  # sumatori d'elements per columnes.

    print("Totals fila:",tot_fila)
    print("Totals columna:",tot_col)

taula = np.array([[2,3,4],
                  [5,6,7]])

taula_2D(taula)
```

```
Totals fila: [ 9 18]
Totals columna: [ 7  9 11]
```

In [237]:

```
#Exercici 4
#Implementa manualment una funció que calculi el coeficient de correlació. Informa't-en
sobre els seus usos i interpretació.

#Un coeficient de correlació es un número que denota la força de la relació entre 2 var
iables.
#Formula del coeficient de correlació:
#la covarianza entre 2 variables dividida por el producte de les desviacions estándar d
e las 2 variables.
#Correlació (r ) = NΣXY - (ΣX)(ΣY) / Sqrt([NΣX 2 - (ΣX) 2 ][NΣY 2 - (ΣY) 2 ])
#Si el coeficient de correlació és positiu, la correlació es directa.
#Si el coeficient de correlació és molt proper a 1 la correlació és molt forta.

import numpy as np
import pandas as pd
#import matplotlib.pyplot as plt
import math

p = np.array([])

def coef_corr(x,y):
    # calculem el coeficient de correlació en diversos moviments.
    media_x = int(sum(x)/len(x)) #Es calcula la mitjana de la primera variable.
    media_y = int(sum(y)/len(y)) #Es calcula la mitjana de la segona variable.

    desv_x = sum(x * x)/len(x) #Es calcula la desviació standard de la primera variabl
e.
    desv_x -= (media_x * media_x)
    desv_x = round(math.sqrt(desv_x),2)

    desv_y = sum(y * y)/len(y) #Es calcula la desviació standard de la segona variable.
    desv_y -= (media_y * media_y)
    desv_y = round(math.sqrt(desv_y),2)

    r = sum(x * y)/len(x * y) # Es calcula la covarianza.
    r = round(r - (media_x * media_y),2)
    r = round(r / (desv_x * desv_y),2)
    print("El coeficient de correlació de ",x,"i",y,"és",r)


t_0 = np.array([2,3,4,4,5,6,6,7,7,8,10,10])
t_1 = np.array([1,3,2,4,4,4,6,4,6,7,9,10])
coef_corr(t_0,t_1)
```

```
El coeficient de correlació de  [ 2  3  4  4  5  6  6  7  7  8 10 10] i [
1  3  2  4  4  4  6  4  6  7  9 10] és 0.94
```

In [ ]:

In [ ]:

In [ ]: