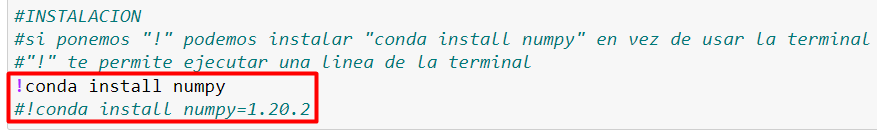
**NUMPY**

Numpy [API reference](https://numpy.org/doc/stable/reference/) → “LA BIBLIA de methods (de numpy)”

**1º Instalar Numpy:**  En la terminal o en el notebook usando **“!”**



**2º Importar la librería:** en el notebok



DEFINICIÓN

Numpy ofrece **gran colección de funciones matemáticas** de alto nivel (es decir, MÉTODOS) para aplicar a **MATRICES MULTIDIMENSIONALES**

* Se pueden usar para representar **datos tabulares**
* Sus matrices son **listas de listas**, donde todos los elementos de una lista son del mismo tipo (*normalmente numéricos*, ya que la razón por la que usa Numpy es para hacer computación numérica). Es decir son MATRICES DE NÚMEROS
* El **size** es **el número total de elementos** en cada lista
* La **shape**  es el **tamaño** de la matriz a lo largo de cada dimensión (ej.:, número de filas y número de columnas para una matriz bidimensional).

**numpy.array(***object*, *dtype=None*, *\**, *copy=True*, *order='K'*, *subok=False*, *ndmin=0*, *like=None***)**

**→ EJ.: np.array(([[1,2,3],[4,5,6]])** → method() de numpy para crear matrices (funciona con list of lists, a list of tuples, a tuple of lists, or a tuple of tuples)

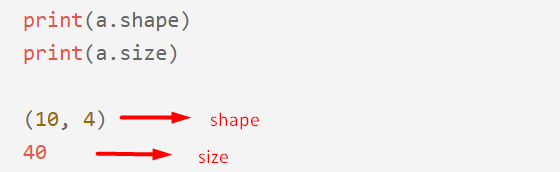
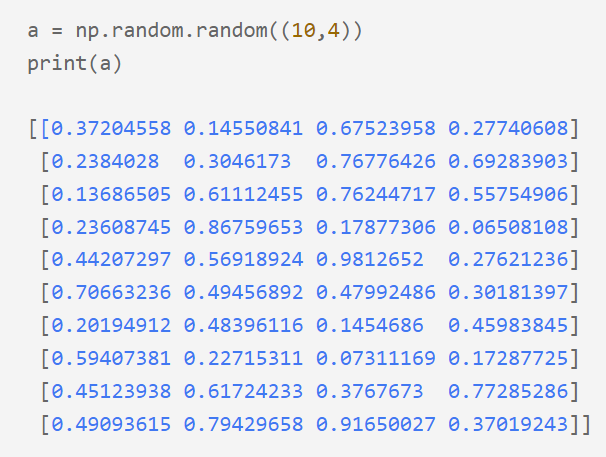
TIPOS DE ARRAYS ***(object)***:

1. unidimensional:

* (8,) → 8 filas,1 columna pero solo con un [], en vez de dos
* (,8) → 1 fila, 8 columna pero solo con un [], en vez de dos

1. **two-dimensional array**

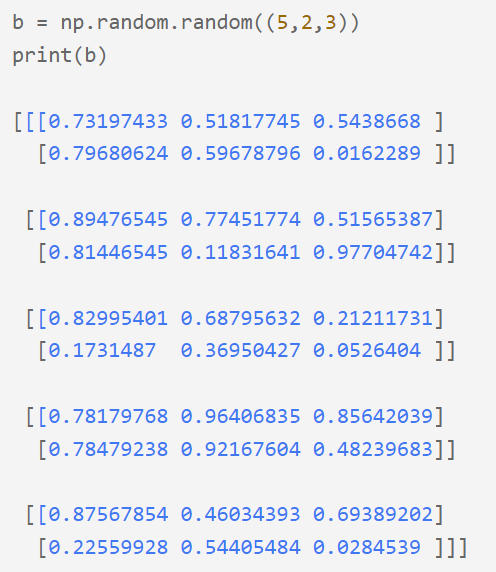
**→ FILAS x COLUMNAS** (los axis siguen el mismo orden empezando por 0)



1. **three-dimensional array**

**→ GRUPOS x FILAS x COLUMNAS**

*(5 grupos de 2x3) (axis=0, axis=1, axis =2)*

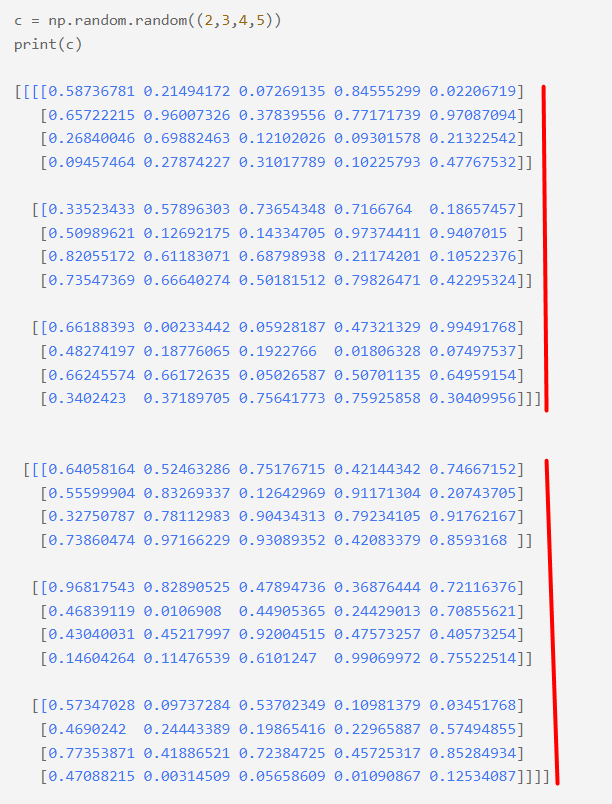
****

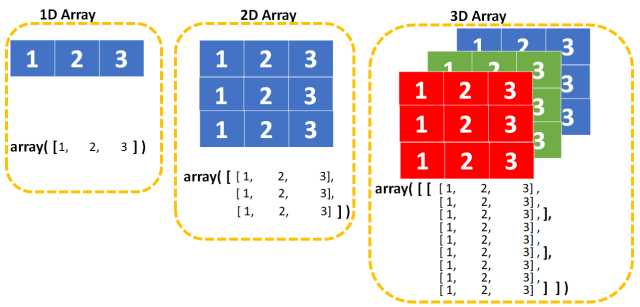
1. **four-dimensional array**

**→ GRUPOS x MATRICES x FILAS x COLUMNAS**

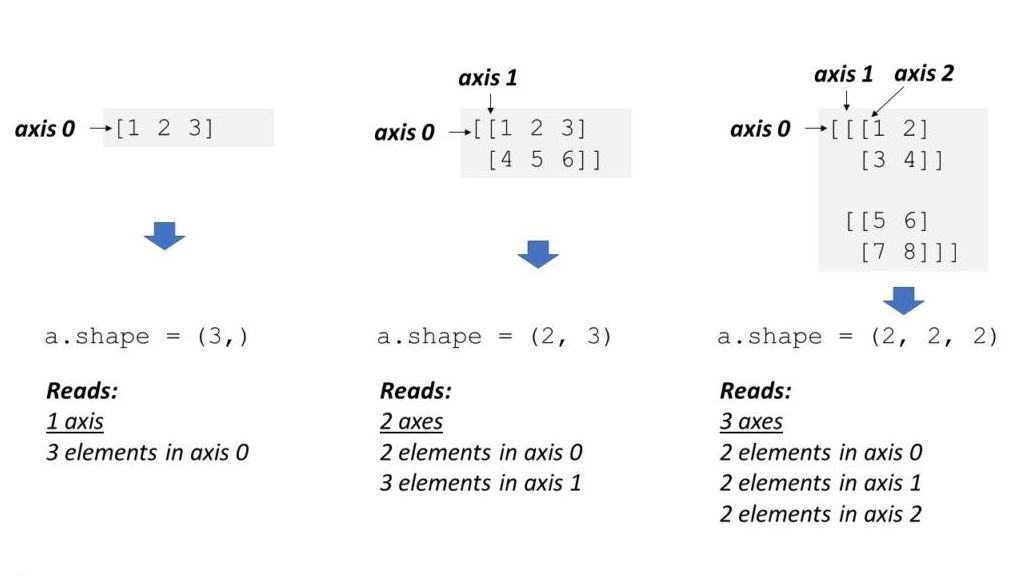
*(2 grupos de 3 matrices de 4x5)(axis=0, axis=1, axis=2, axis=3)*

***→ 2 elem. en el axis=0, 3 en el axis=1, 4 en el axis=2, 5 en el axis =3***

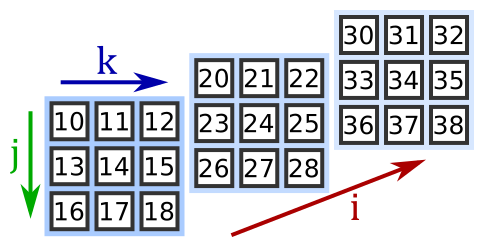




---------------------------------------------------------------------------------------------------------------------------



---------------------------------------------------------------------------------------------------------------------------

****

**Extracting Data from Arrays**

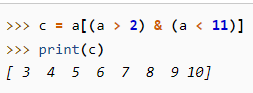
Necesitamos **hacer referencia a** los **índices de los valores** que queremos extraer:

**→** si es un **número solo,** es el elemento en concreto **[index-1]**

**→** si hay **“:” [index:index-1]** (es como decir [de la x a la x] del mismo elemento)

**→** se pueden poner **condiciones en cada index**:





**→** cuando hay **solo “:”** quiere decir también que son **todos los elementos.**

**Ej mezcla.: *nombre\_matriz[****index-1,* ***:****, index:index-1, index-1****]***

**=** *nombre\_matriz****[2****,* ***:****,* ***2:4****,* ***5]***

***= grupo 2, todas las matrices, fila 2 y 3, columna 5***

**→** Siempre va de **IZQUIERDA a DERECHA**, hasta llegar a FILAxCOLUMNA:

* GRUPOS x MATRICES x FILAS x COLUMNAS

→ matriz[0] ===== dame el grupo 0

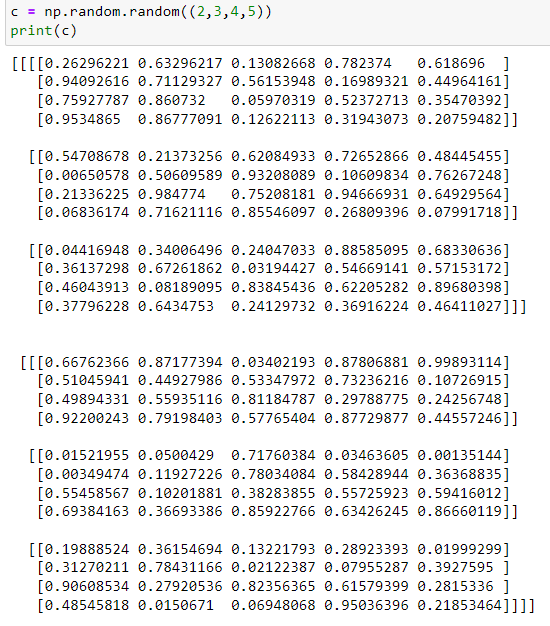
→ matriz[0,1] ===== dame el grupo 0, matriz 1

→ matriz[0,1,2] ===== dame el grupo 0, matriz 1, fila 2

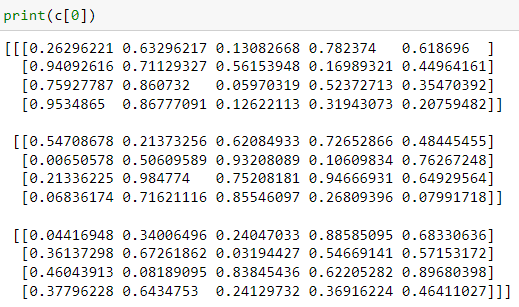
→ matriz[0,1,2:4] ===== dame el grupo 0, matriz 1, fila 2 y 3

* MATRICES x FILAS x COLUMNAS
* FILAS x COLUMNAS
* COLUMNAS ¡NO! (el último elemento siempre es FILAxCOLUMNA, sino no es un array)

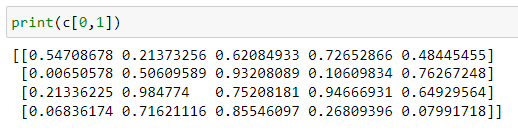
**EJEMPLO:** creamos matriz = np.random.random((2,3,4,5))



* **MATRIZ:** grupo 0

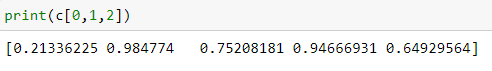


* **MATRIZ:** grupo 0, matriz 1 entera

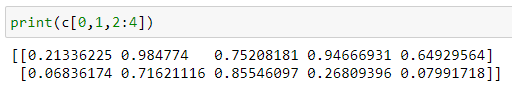


* **FILA:** grupo 0, matriz 1 entera

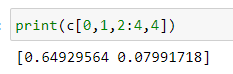
1. fila 2 entera



1. fila 2 y 3 enteras

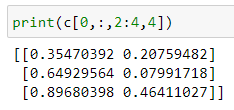


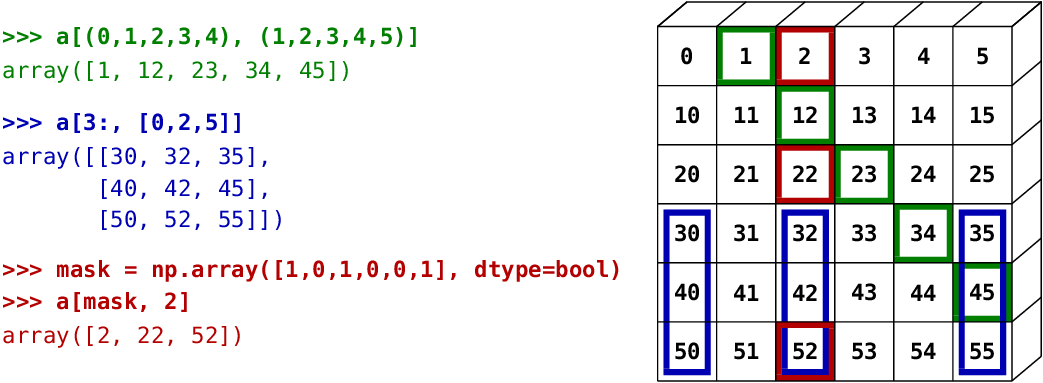
* **COLUMNA:** grupo 0, matriz 1, fila 2 y 3, columna 4 enteras



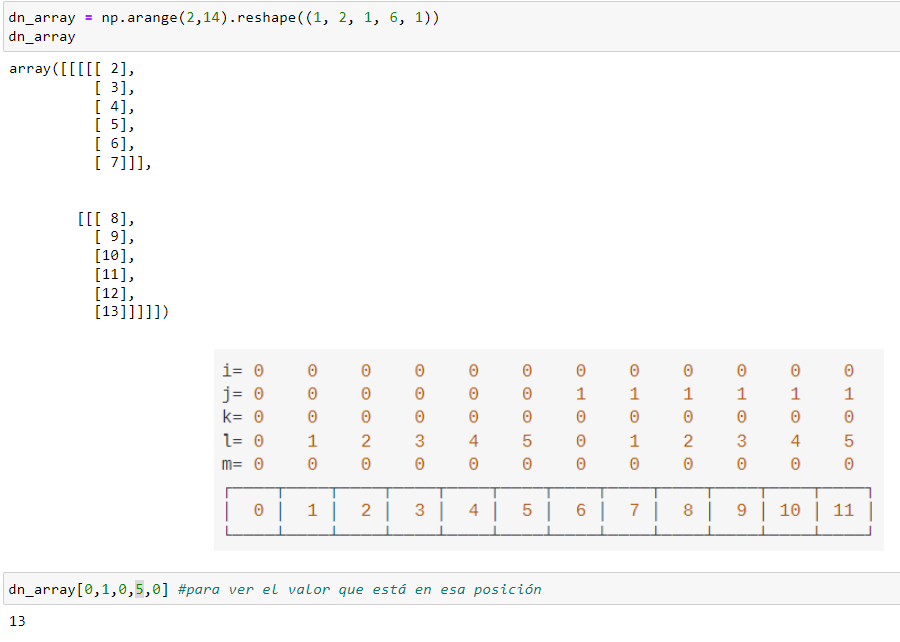
**→ nombre\_matriz[index-1**, **index-1**, **index:index-1**, **index-1]**

* **MIX:** grupo 0, todas las matrices, filas 2 y 3, columna 4

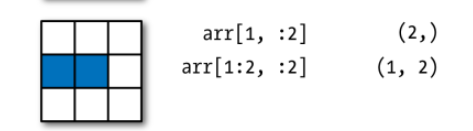
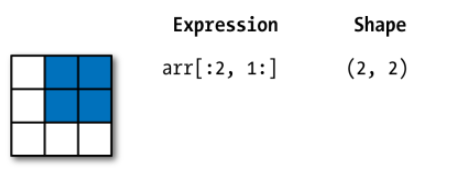
****

→ Si se pone entre **paréntesis o corchetes** el index que quieres, se puede indicar las **filas o columnas** concretas****

---------------------------------------------------------------------------------------------------------------------------

****

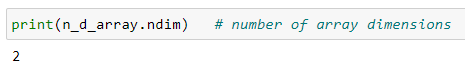
---------------------------------------------------------------------------------------------------------------------------

****

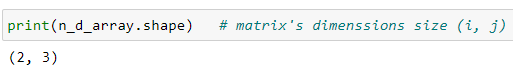
**Array Object Attributes**

****

* name\_array**.ndim**:número de DIMENSIONES



* name\_array**.shape**:GRUPOS x MATRICES x FILAS x COLUMNAS



* name\_array**.size**:Total de elementos



* name\_array**.dtype**:Type of elements + size



* name\_array**.itemsize**:length en bytes (useless para nosotros)



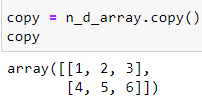
**Array Object Methods**

→ [listado entero](https://numpy.org/doc/stable/reference/arrays.ndarray.html#array-methods)

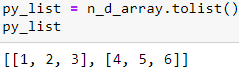
Se aplican al NOMBRE/OBJETO del array. También vale **np.method(objeto)**

****

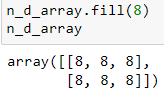
* name\_array**.copy()**:crea una copia



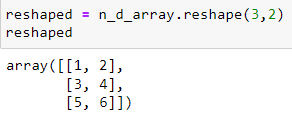
* name\_array**.tolist()**:crea una list



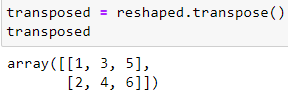
* name\_array**.fill()**:rellena un array con lo que le digas



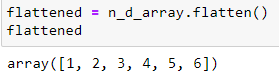
* name\_array**.reshape()**:devuelve un array con otra forma



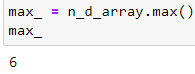
* name\_array**.transpose()**:transpone la matriz

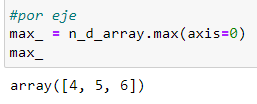


* name\_array**.flatten()**:lo aplana en una línea

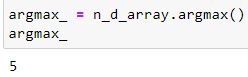


* name\_array**.max()**:devuelve el VALOR MÁXIMO

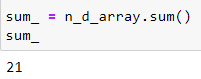




* name\_array**.argmax()**:devuelve el ÍNDICE del valor máximo



* name\_array**.sum()**:SUMA todos los elementos



**NUMPY METHODS() / ROUTINES**

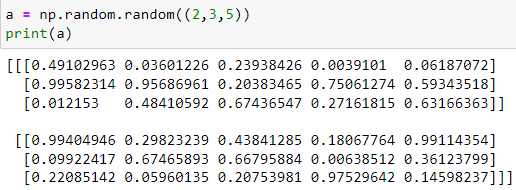
→ [listado entero](https://numpy.org/doc/stable/reference/routines.html)

1. [array creations routines](https://numpy.org/doc/stable/reference/routines.array-creation.html)
2. [array manipulation routines](https://numpy.org/doc/stable/reference/routines.array-manipulation.html)

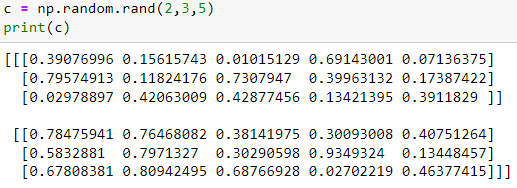
* **np.array(([[1,2,3],[4,5,6]])**:crea matrices (funciona con list of lists, a list of tuples, a tuple of lists, or a tuple of tuples)

****

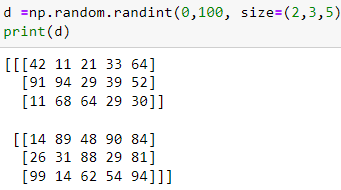
* **np.random.random((2,3,5))**:valores random en el shape que se le indique



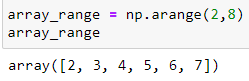
* **np.random.rand(2,3,5)**:igual



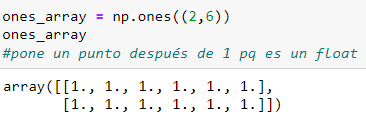
* **np.random.randint(0,100, size=(2,3,5)):** intervalo random por size



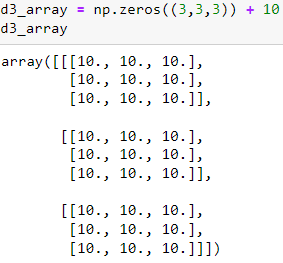
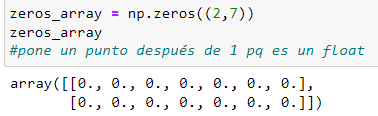
* **np.arange()**:devuelve valores en un intervalo



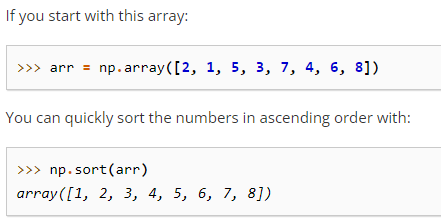
* **np.ones()**:rellena de “1”



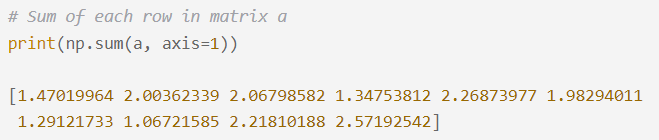
* **np.zeros()**:rellena de “0”

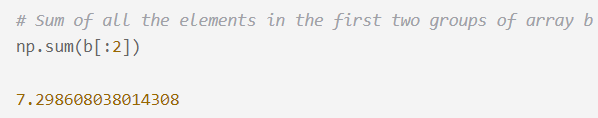


* **np.sort()**:ordena el array

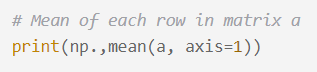


* **np.sum()**:**SUMA** de cualquier **elemento** que seleccione de una matriz.

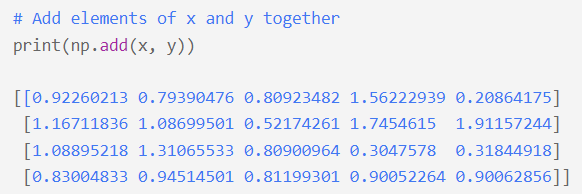




* **np.mean()**:**MEDIA** de unos elementos concretos



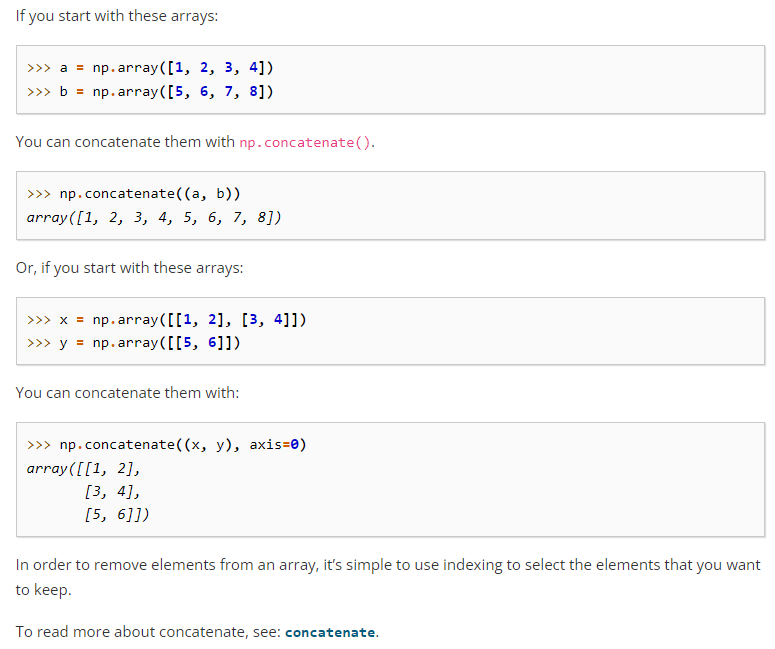
* **np.add()**:**SUMA** una matriz con otra



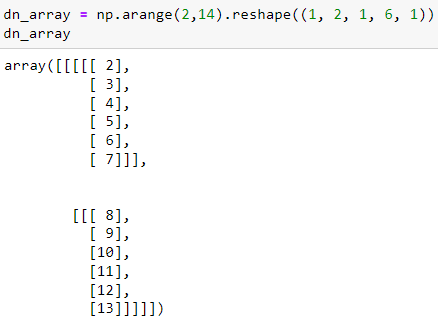
* **np.where(condición,** resultado si cumple la cond.**,** resultado si no cumple la cond.**)**



* **np.subtract()**:**RESTA** una **matriz** con otra
* **np.multiply()**:**MULTIPLICA** una **matriz** con otra
* **np.divide()**:**DIVIDE** una **matriz** con otra
* **np.nditer()**:**ITERA** todos los elementos de cualquier **matriz**
* **np.concatenate()**:concatena (es fácil a través del index, concatenar y dejar fuera elementos, como si hiceses un remove de ellos):

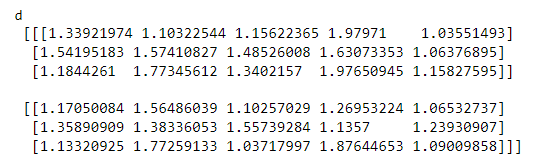


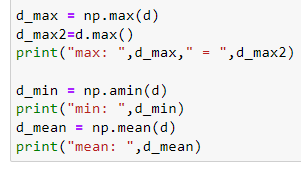
→ se pueden **mezclar**:

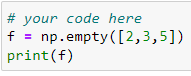


**EJERCICIO DE INTERACCIÓN:**

* Las variables son:

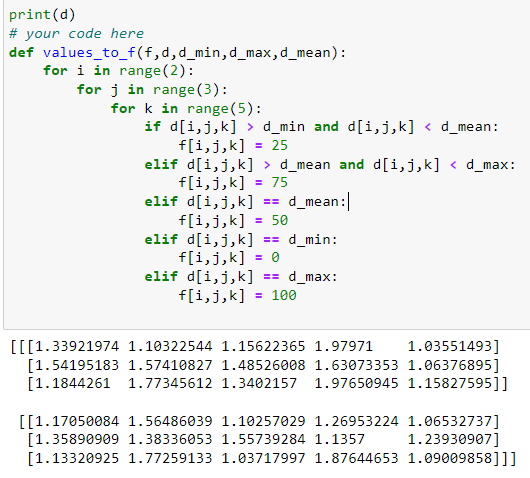




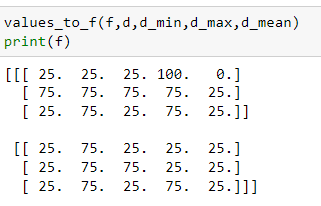


* queremos sustituir en f por 0, 25, 50, 75 y 100 según condiciones:

*→ 1º iteramos los dos grupos, 2º iteramos las filas, 3º iteramos las columnas (así obtendremos el index final para modificar el array a través de “slicing”)*



* resultado:



\* MÁS FÁCIL EL EJERCICIO ANTERIOR:

