Juan David Alvarez cc 1112791148

APILAMIENTO

In [2]:
```
import numpy as np

a = np.arange(9).reshape(3,3)
print('a =\n', a, '\n')

b = a*2
print('b =\n', b)
```

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

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]]
```

In [3]:
```
print('a =\n', a, '\n')
print('b =\n', b, '\n')

print('Apilamiento horizontal =\n', np.hstack((a,b)) )
```

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

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]]

Apilamiento horizontal =
 [[ 0  1  2  0  2  4]
 [ 3  4  5  6  8 10]
 [ 6  7  8 12 14 16]]
```

In [4]:
```
print('a =\n', a, '\n')
print('b =\n', b, '\n')

print( 'Apilamiento horizontal con concatenate = \n',
np.concatenate((a,b), axis=1) )
```

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

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]]

Apilamiento horizontal con concatenate =
 [[ 0  1  2  0  2  4]
 [ 3  4  5  6  8 10]
 [ 6  7  8 12 14 16]]
```

In [5]:
```
print('a =\n', a, '\n')
print('b =\n', b, '\n')
print( 'Apilamiento vertical =\n', np.vstack((a,b)) )
```

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

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]]

Apilamiento vertical =
 [[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 0  2  4]
 [ 6  8 10]
 [12 14 16]]
```

In [6]:
```
print('a =\n', a, '\n')
print('b =\n', b, '\n')

print( 'Apilamiento vertical con concatenate =\n',
np.concatenate((a,b), axis=0) )
```

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

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]]

Apilamiento vertical con concatenate =
 [[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 0  2  4]
 [ 6  8 10]
 [12 14 16]]
```

In [7]:
```
print('a =\n', a, '\n')
print('b =\n', b, '\n')

print( 'Apilamiento en profundidad =\n', np.dstack((a,b)) )
```

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

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]]

Apilamiento en profundidad =
 [[[ 0  0]
  [ 1  2]
  [ 2  4]]

 [[ 3  6]
  [ 4  8]
  [ 5 10]]

 [[ 6 12]
  [ 7 14]
  [ 8 16]]]
```

In [8]:
```
print('a =\n', a, '\n')
print('b =\n', b, '\n')

print( 'Apilamiento por columnas =\n',
np.column_stack((a,b)) )
```

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

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]]

Apilamiento por columnas =
 [[ 0  1  2  0  2  4]
 [ 3  4  5  6  8 10]
 [ 6  7  8 12 14 16]]
```

In [9]:
```
print('a =\n', a, '\n')
print('b =\n', b, '\n')

print( 'Apilamiento por filas =\n',
np.row_stack((a,b)) )
```

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

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]]

Apilamiento por filas =
 [[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 0  2  4]
 [ 6  8 10]
 [12 14 16]]
```

DIVISIÓN DE ARRAYS

In [10]:
```
print(a, '\n')
print('Array con división horizontal =\n', np.hsplit(a, 3), '\n')

print('Array con división horizontal, uso de split() =\n',
np.split(a, 3, axis=1))
```

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

Array con división horizontal =
 [array([[0],
       [3],
       [6]]), array([[1],
       [4],
       [7]]), array([[2],
       [5],
       [8]])]

Array con división horizontal, uso de split() =
 [array([[0],
       [3],
       [6]]), array([[1],
       [4],
       [7]]), array([[2],
       [5],
       [8]])]
```

In [11]:
```
print(a, '\n')

print('División Vertical = \n', np.vsplit(a, 3), '\n')
print('Array con división vertical, uso de split() =\n',
np.split(a, 3, axis=0))
```

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

División Vertical =
 [array([[0, 1, 2]]), array([[3, 4, 5]]), array([[6, 7, 8]])]

Array con división vertical, uso de split() =
 [array([[0, 1, 2]]), array([[3, 4, 5]]), array([[6, 7, 8]])]
```

In [12]:
```
c = np.arange(27).reshape(3, 3, 3)
print(c, '\n')
print('División en profundidad =\n', np.dsplit(c,3), '\n')
```

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

 [[ 9 10 11]
  [12 13 14]
  [15 16 17]]

 [[18 19 20]
  [21 22 23]
  [24 25 26]]]

División en profundidad =
 [array([[[ 0],
         [ 3],
         [ 6]],

        [[ 9],
         [12],
         [15]],

        [[18],
         [21],
         [24]]]), array([[[ 1],
         [ 4],
         [ 7]],

        [[10],
         [13],
         [16]],

        [[19],
         [22],
         [25]]]), array([[[ 2],
         [ 5],
         [ 8]],

        [[11],
         [14],
         [17]],

        [[20],
         [23],
         [26]]])]
```

PROPIEDADES DE LOS ARRAYS

In [13]:
```
print(b, '\n')
print('ndim: ', b.ndim)
```

```
[[ 0  2  4]
 [ 6  8 10]
 [12 14 16]]

ndim:  2
```

In [14]:
```
print(b, '\n')
print('size: ', b.size)
```

```
[[ 0  2  4]
 [ 6  8 10]
 [12 14 16]]

size:  9
```

In [15]:
```
print('itemsize: ', b.itemsize)
```

```
itemsize:  4
```

In [16]:
```
print(b, '\n')
print('nbytes: ', b.nbytes, '\n')

print('nbytes equivalente: ', b.size * b.itemsize)
```

```
[[ 0  2  4]
 [ 6  8 10]
 [12 14 16]]

nbytes:  36

nbytes equivalente:  36
```

In [17]:
```
b.resize(6,4)
print(b, '\n')
print('Transpuesta: ', b.T)
```

```
[[ 0  2  4  6]
 [ 8 10 12 14]
 [16  0  0  0]
 [ 0  0  0  0]
 [ 0  0  0  0]
 [ 0  0  0  0]]

Transpuesta:  [[ 0  8 16  0  0  0]
 [ 2 10  0  0  0  0]
 [ 4 12  0  0  0  0]
 [ 6 14  0  0  0  0]]
```

In [18]:
```
b = np.array([1.j + 1, 2.j + 3])
print('Complejo: \n', b)
```

```
Complejo:
 [1.+1.j 3.+2.j]
```

In [19]:
```
print('real: ', b.real, '\n')
print('imaginario: ', b.imag)
```

```
real:  [1. 3.]

imaginario:  [1. 2.]
```

In [20]:
```
print(b.dtype)
```

```
complex128
```

In [21]:
```
b = np.arange(4).reshape(2,2)
print(b, '\n')
f = b.flat
print(f, '\n')

for item in f: print (item)

print('\n')
print('Elemento 2: ', b.flat[2])

b.flat = 7
print(b, '\n')
b.flat[[1,3]] = 1
print(b, '\n')
```

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

<numpy.flatiter object at 0x000001AC43093D00>

0
1
2
3


Elemento 2:  2
[[7 7]
 [7 7]]

[[7 1]
 [7 1]]
```