In [106]:

```
print('veronica palacio villada codigo 1192808282\n')
print('ingenieria de sistemas y computación  \n')
print('computacion blanda\n')
```

```
veronica palacio villada codigo 1192808282

ingenieria de sistemas y computación

computacion blanda
```

In [107]:

```


import numpy as np
a = np.arange(12).reshape(4,3)
print('a =\n', a, '\n')
b = a*2
print('b =\n', b)

```

```
a =
 [[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]]

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]
```

In [108]:

```
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]
 [ 9 10 11]]

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]

Apilamiento horizontal =
 [[ 0  1  2  0  2  4]
 [ 3  4  5  6  8 10]
 [ 6  7  8 12 14 16]
 [ 9 10 11 18 20 22]]
```

In [109]:

```
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]
 [ 9 10 11]]

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]

Apilamiento horizontal con concatenate =
 [[ 0  1  2  0  2  4]
 [ 3  4  5  6  8 10]
 [ 6  7  8 12 14 16]
 [ 9 10 11 18 20 22]]
```

In [110]:

```
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]
 [ 9 10 11]]

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]

Apilamiento vertical =
 [[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]
 [ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]
```

In [111]:

```
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]
 [ 9 10 11]]

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]

Apilamiento vertical con concatenate =
 [[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]
 [ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]
```

In [112]:

```
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]
 [ 9 10 11]] 

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]] 

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

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

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

 [[ 9 18]
  [10 20]
  [11 22]]]
```

In [113]:

```
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]
 [ 9 10 11]]

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]

Apilamiento por columnas =
 [[ 0  1  2  0  2  4]
 [ 3  4  5  6  8 10]
 [ 6  7  8 12 14 16]
 [ 9 10 11 18 20 22]]
```

In [114]:

```
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]
 [ 9 10 11]]

b =
 [[ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]

Apilamiento por filas =
 [[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]
 [ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]
```

In [115]:

```
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]
 [ 9 10 11]]

Array con división horizontal =
 [array([[0],
       [3],
       [6],
       [9]]), array([[ 1],
       [ 4],
       [ 7],
       [10]]), array([[ 2],
       [ 5],
       [ 8],
       [11]])]

Array con división horizontal, uso de split() =
 [array([[0],
       [3],
       [6],
       [9]]), array([[ 1],
       [ 4],
       [ 7],
       [10]]), array([[ 2],
       [ 5],
       [ 8],
       [11]])]
```

In [116]:

```
print(a, '\n')
print('División Vertical = \n', np.vsplit(a, 4), '\n')
print('Array con división vertical, uso de split() =\n',
np.split(a, 4, axis=0))
```

```
[[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]]

División Vertical =
 [array([[0, 1, 2]]), array([[3, 4, 5]]), array([[6, 7, 8]]), array([[ 9, 1
0, 11]])]

Array con división vertical, uso de split() =
 [array([[0, 1, 2]]), array([[3, 4, 5]]), array([[6, 7, 8]]), array([[ 9, 1
0, 11]])]
```

In [117]:

```
c = np.arange(150).reshape(25,2,3)
print(c, '\n')
print('División en profundidad =\n', np.dsplit(c,3), '\n')
```

```
         [ 41]],

        [[ 44],
         [ 47]],

        [[ 50],
         [ 53]],

        [[ 56],
         [ 59]],

        [[ 62],
         [ 65]],

        [[ 68],
         [ 71]],

        [[ 74],
         [ 77]],
```

In [118]:

```
print(b, '\n')
print('ndim: ', b.ndim)
```

```
[[ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]

ndim:  2
```

In [119]:

```
print(b, '\n')
print('size: ', b.size)
```

```
[[ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]

size:  12
```

In [120]:

```
print('itemsize: ', b.itemsize)
```

```
itemsize:  4
```

In [121]:

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

```
[[ 0  2  4]
 [ 6  8 10]
 [12 14 16]
 [18 20 22]]

nbytes:  48

nbytes equivalente:  48
```

In [122]:

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

```
[[ 0  2  4  6]
 [ 8 10 12 14]
 [16 18 20 22]
 [ 0  0  0  0]
 [ 0  0  0  0]
 [ 0  0  0  0]]

Transpuesta:  [[ 0  8 16  0  0  0]
 [ 2 10 18  0  0  0]
 [ 4 12 20  0  0  0]
 [ 6 14 22  0  0  0]]
```

In [123]:

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

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

In [124]:

```
print('real: ', b.real, '\n')
print('imaginario: ', b.imag)
```

```
real:  [1. 3.]

imaginario:  [1. 2.]
```

In [125]:

```
print(b.dtype)
```

```
complex128
```

In [126]:

```
b = np.arange(10).reshape(2,5)
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 4]
 [5 6 7 8 9]]

<numpy.flatiter object at 0x0000027C616A1C60>

0
1
2
3
4
5
6
7
8
9


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

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

In [ ]:

In [ ]: