

```
In [1]: import numpy as np
```

```
In [2]: np.__version__
```

```
Out[2]: '2.3.5'
```

```
In [5]: import numpy as np
```

```
arr = np.array([1,2,3,4]) # Plus 2  
print(arr + 2)
```

```
[3 4 5 6]
```

```
In [7]: arr = np.array([2,4,6])  
print(arr * 2)
```

```
[ 4  8 12]
```

## Creating Array

```
In [8]: my_list = [0,1,2,3,4,5]  
my_list
```

```
Out[8]: [0, 1, 2, 3, 4, 5]
```

```
In [9]: type(my_list)
```

```
Out[9]: list
```

```
In [10]: arr = np.array(my_list)
```

```
In [11]: arr
```

```
Out[11]: array([0, 1, 2, 3, 4, 5])
```

```
In [12]: type(arr)
```

```
Out[12]: numpy.ndarray
```

## arange()

```
In [13]: np.arange(15) # it will generate number sequence wise
```

```
Out[13]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14])
```

```
In [14]: np.arange(20)
```

```
Out[14]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
               17, 18, 19])
```

```
In [15]: np.arange(5.0) # now generate number in float
```

```
Out[15]: array([0., 1., 2., 3., 4.])
```

```
In [16]: np.arange(10, 50, 5)  #(staring no. ,end no. , gap,step count)
```

```
Out[16]: array([10, 15, 20, 25, 30, 35, 40, 45])
```

```
In [17]: np.arange(2,10)
```

```
Out[17]: array([2, 3, 4, 5, 6, 7, 8, 9])
```

```
In [18]: np.arange(20,10)
```

```
Out[18]: array([], dtype=int64)
```

```
In [19]: np.arange(20,10,-1) # reverse order
```

```
Out[19]: array([20, 19, 18, 17, 16, 15, 14, 13, 12, 11])
```

```
In [20]: np.arange(-20,10)
```

```
Out[20]: array([-20, -19, -18, -17, -16, -15, -14, -13, -12, -11, -10, -9, -8,
               -7, -6, -5, -4, -3, -2, -1,  0,  1,  2,  3,  4,  5,
                6,  7,  8,  9])
```

```
In [21]: np.arange(30,20) # first arg always smaller than second arg
```

```
Out[21]: array([], dtype=int64)
```

```
In [22]: np.arange(10,30,5) # 10- starting from 30- end point 5 - step count
```

```
Out[22]: array([10, 15, 20, 25])
```

## zeros()

```
In [23]: np.zeros(3) #parameter tuning
```

```
Out[23]: array([0., 0., 0.])
```

```
In [26]: np.zeros(5)
```

```
Out[26]: array([0., 0., 0., 0., 0.])
```

```
In [27]: np.zeros(5, dtype=int) #hyperparameter tuning
```

```
Out[27]: array([0, 0, 0, 0, 0])
```

```
In [28]: np.zeros((2,2)) # float matrix
```

```
Out[28]: array([[0., 0.],
               [0., 0.]])
```

```
In [29]: np.zeros((2,2), dtype=int) # int matrix
```

```
Out[29]: array([[0, 0],
               [0, 0]])
```

```
In [30]: np.zeros((3,5)) # 3 rows , 5 columns
```

```
Out[30]: array([[0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0.]])
```

```
In [31]: np.zeros((10,15))
```

```
Out[31]: array([[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]])
```

```
In [32]: np.zeros((5,10))
```

```
Out[32]: array([[0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
               [0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]])
```

```
In [33]: np.zeros((8,4))
```

```
Out[33]: array([[0., 0., 0., 0.],
               [0., 0., 0., 0.],
               [0., 0., 0., 0.],
               [0., 0., 0., 0.],
               [0., 0., 0., 0.],
               [0., 0., 0., 0.],
               [0., 0., 0., 0.],
               [0., 0., 0., 0.]])
```

```
In [34]: np.zeros((8,4), dtype=int)
```

```
Out[34]: array([[0, 0, 0, 0],
                [0, 0, 0, 0],
                [0, 0, 0, 0],
                [0, 0, 0, 0],
                [0, 0, 0, 0],
                [0, 0, 0, 0],
                [0, 0, 0, 0],
                [0, 0, 0, 0]])
```

## ones()

```
In [36]: np.ones(3)
```

```
Out[36]: array([1., 1., 1.])
```

```
In [38]: np.ones((3),dtype=int)
```

```
Out[38]: array([1, 1, 1])
```

```
In [41]: np.ones((5,7),dtype=int)
```

```
Out[41]: array([[1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1]])
```

```
In [40]: np.ones((5,4),dtype=int)
```

```
Out[40]: array([[1, 1, 1, 1],
                [1, 1, 1, 1],
                [1, 1, 1, 1],
                [1, 1, 1, 1],
                [1, 1, 1, 1]])
```

```
In [42]: np.random.rand(5) # their output will be changing every time
```

```
Out[42]: array([0.40542944, 0.4802198 , 0.45252924, 0.30696795, 0.42916635])
```

```
In [44]: np.random.randint(5)
```

```
Out[44]: 0
```

```
In [46]: np.random.rand(5,2)
```

```
Out[46]: array([[0.81763296, 0.82088062],
                [0.23065036, 0.65157438],
                [0.21679086, 0.72648365],
                [0.6761499 , 0.45442544],
                [0.06323856, 0.97619646]])
```

```
In [54]: np.random.randint(2,20)
```

Out[54]: 16

In [56]: `np.random.randint(1,6,4) # ( from no. , to this number , but only 4 number print)`

Out[56]: `array([1, 5, 2, 5], dtype=int32)`

In [60]: `np.random.randint(30,20,10)`

```
-----  
ValueError                                Traceback (most recent call last)  
Cell In[60], line 1  
----> 1 np.random.randint(30,20,10)  
  
File numpy/random/mtrand.pyx:794, in numpy.random.mtrand.RandomState.randint()  
  
File numpy/random/_bounded_integers.pyx:1425, in numpy.random._bounded_integers._rand_int32()  
  
ValueError: low >= high
```

In [62]: `np.random.randint(-30,20,10)`

Out[62]: `array([-17, 1, -6, 8, 14, -17, 19, -16, -6, 14], dtype=int32)`

In [65]: `np.random.randint(10,20,(4,4))`

Out[65]: `array([[11, 19, 14, 19],  
 [12, 10, 10, 16],  
 [18, 13, 11, 11],  
 [16, 14, 15, 12]], dtype=int32)`

In [66]: `np.arange(1,13)`

Out[66]: `array([ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])`

In [68]: `np.arange(1,13).reshape(3,4)`

Out[68]: `array([[ 1, 2, 3, 4],  
 [ 5, 6, 7, 8],  
 [ 9, 10, 11, 12]])`

In [69]: `np.arange(20,30)`

Out[69]: `array([20, 21, 22, 23, 24, 25, 26, 27, 28, 29])`

In [74]: `np.arange(20,36).reshape(4,4)`

Out[74]: `array([[20, 21, 22, 23],  
 [24, 25, 26, 27],  
 [28, 29, 30, 31],  
 [32, 33, 34, 35]])`

In [75]: `np.arange(1,10)`

```
Out[75]: array([1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
In [77]: np.arange(1,13).reshape(3,4)
```

```
Out[77]: array([[ 1,  2,  3,  4],
                [ 5,  6,  7,  8],
                [ 9, 10, 11, 12]])
```

```
In [78]: np.arange(1,13).reshape(12, 1)
```

```
Out[78]: array([[ 1],
                [ 2],
                [ 3],
                [ 4],
                [ 5],
                [ 6],
                [ 7],
                [ 8],
                [ 9],
                [10],
                [11],
                [12]])
```

```
In [79]: np.arange(1,15).reshape(14,1)
```

```
Out[79]: array([[ 1],
                [ 2],
                [ 3],
                [ 4],
                [ 5],
                [ 6],
                [ 7],
                [ 8],
                [ 9],
                [10],
                [11],
                [12],
                [13],
                [14]])
```

```
In [80]: b = np.random.randint(10,20,(5,4))
b
```

```
Out[80]: array([[10, 16, 14, 18],
                [15, 14, 14, 12],
                [17, 17, 15, 18],
                [10, 16, 10, 14],
                [13, 16, 19, 11]], dtype=int32)
```

```
In [81]: type(b)
```

```
Out[81]: numpy.ndarray
```

```
In [82]: b[:]
```

```
Out[82]: array([[10, 16, 14, 18],
               [15, 14, 14, 12],
               [17, 17, 15, 18],
               [10, 16, 10, 14],
               [13, 16, 19, 11]], dtype=int32)
```

```
In [83]: b[1:3]
```

```
Out[83]: array([[15, 14, 14, 12],
               [17, 17, 15, 18]], dtype=int32)
```

```
In [84]: b[1,2]
```

```
Out[84]: np.int32(14)
```

```
In [85]: b
```

```
Out[85]: array([[10, 16, 14, 18],
               [15, 14, 14, 12],
               [17, 17, 15, 18],
               [10, 16, 10, 14],
               [13, 16, 19, 11]], dtype=int32)
```

```
In [88]: b[1,3]
```

```
Out[88]: np.int32(12)
```

```
In [89]: b[0,2]
```

```
Out[89]: np.int32(14)
```

```
In [90]: b[1,-1]
```

```
Out[90]: np.int32(12)
```

```
In [91]: b[1,-2]
```

```
Out[91]: np.int32(14)
```

```
In [92]: b[2,-3]
```

```
Out[92]: np.int32(17)
```

```
In [93]: np.random.randint(10,20,(4,4))
```

```
Out[93]: array([[14, 10, 14, 18],
               [10, 11, 16, 10],
               [13, 10, 15, 19],
               [18, 13, 11, 12]], dtype=int32)
```

```
In [94]: b[-4,-2]
```

```
Out[94]: np.int32(14)
```

```
In [95]: b[0,-2]
```

```
Out[95]: np.int32(14)
```

```
In [96]: b[0,-3]
```

```
Out[96]: np.int32(16)
```

```
In [97]: b
```

```
Out[97]: array([[10, 16, 14, 18],  
               [15, 14, 14, 12],  
               [17, 17, 15, 18],  
               [10, 16, 10, 14],  
               [13, 16, 19, 11]], dtype=int32)
```

```
In [101... b[-4,2]
```

```
Out[101... np.int32(14)
```

```
In [100... b[-1,2]
```

```
Out[100... np.int32(19)
```

```
In [ ]:
```