```
In [1]: import numpy as np
In [2]: deepa = np.array([1,2,3,4,5,6])
In [3]: type(deepa)
Out[3]: numpy.ndarray
In [4]: print(type(deepa))
         <class 'numpy.ndarray'>
In [5]: import numpy as np
         x = np.array([5, 8,
                       9, 10,
                       111)
         y=12
         type(x)
         print(y)
         12
In [8]: y = np.array([[1,2,3],[4,5,6],[7,8,9]])
         print(y)
         [[1 2 3]
          [4 5 6]
          [7 8 9]]
In [9]: print(y.ndim,y.size,y.dtype,y.shape)
         2 9 int32 (3, 3)
In [10]: import numpy as np
```



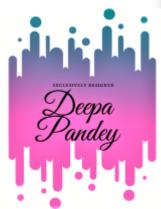
```
x = np.array([[3.2, 7.8, 9.2],
                      [4.5, 9.1, 1.2]], dtype='int64')
         print(x.itemsize)
         8
In [11]: import numpy as np
         y = np.array([3+4j, 0.4+7.8j])
         print(y.dtype)
         complex128
In [3]: print(y.shape)
         NameError
                                                   Traceback (most recent call l
         ast)
         <ipython-input-3-a2d3d6726c7a> in <module>
         ---> 1 print(y.shape)
         NameError: name 'y' is not defined
In [18]: print(y.size)
         2
In [19]: n = [5, 10, 15, 20, 25]
         x = np.array(n)
         type(x)
Out[19]: numpy.ndarray
In [20]: print(x.ndim,x.shape,x.size)
         1 (5,) 5
```



```
In [22]: n = [[-1, -2, -3, -4], [-2, -4, -6, -8]]
         y = np.array(n)
         type(y)
Out[22]: numpy.ndarray
In [24]: print(y.ndim,y.shape,y.size,y.dtype,y.itemsize)
         2 (2, 4) 8 int32 4
In [26]: ab = np.full(shape=(3,4,5),fill_value=2)
         print(ab)
         [[[2 2 2 2 2]
           [2 2 2 2 2]
           [2 2 2 2 2]
           [2 2 2 2 2]]
          [[2 2 2 2 2]
           [2 2 2 2 2]
           [2 2 2 2 2]
           [2 2 2 2 2]]
          [[2 2 2 2 2]
           [2 2 2 2 2]
           [2 2 2 2 2]
           [2 2 2 2 2]]]
In [27]: abc = np.zeros(shape=(3,2,1))
         print(abc)
         [[[0.]
           [0.]]
          [[0.]
           [0.]]
```



```
[[0.]]
           [0.11]
In [38]: #arange : Numbers created based on step value.
         ##Syntax - numpy.arange([start, ]stop, [step, ]dtype=None)
In [29]: abcd = np.arange(10,20,2)
         print(abcd)
         [10 12 14 16 18]
In [35]: #linspace : Numbers created based on size value.
         ##Syntax - numpy.linspace(start, stop, #num inbetween, endpoint=True, r
         etstep=False, dtype=None)
In [31]: abcde = np.linspace(10,100,9)
         print(abcde)
         [ 10.
                                43.75 55.
                                              66.25 77.5
                                                            88.75 100. ]
                  21.25 32.5
In [43]: np.random.seed(100)
         xyz = np.random.rand(3)
         print(xyz)
         [0.54340494 0.27836939 0.42451759]
In [44]: np.random.seed(100)
         xyz = np.random.randint(1,100,3)
         print(xyz)
         [ 9 25 68]
In [49]: from io import StringIO
         import numpy as np
         x = StringIO('''88.25 \$ 93.45 \$ 72.60 \$ 90.90
```



```
72.3 $ 78.85 $ 92.15 $ 65.75
         90.5 $ 92.45 $ 89.25 $ 94.50
         111)
         d = np.loadtxt(x,delimiter=' $ ')
         print(d)
         print(d.ndim, d.shape)
         [[88.25 93.45 72.6 90.9]
          [72.3 78.85 92.15 65.75]
          [90.5 92.45 89.25 94.5 ]]
         2(3, 4)
In [50]: import numpy as np
         print(np.array(([1, 2], (3,4))).shape)
         (2, 2)
In [51]: import numpy as np
         z = np.eye(2)
         print(z)
         [[1. 0.]
          [0. 1.]]
In [56]: x1 = [[[-1,1],[-2,2]],[[-3,3],[-4,4]]]
         xx=np.array(x1)
         print(xx.ndim,xx.shape,xx.size)
         3 (2, 2, 2) 8
In [59]: x = np.full(shape=(3,3,3),fill value=1)
Out[59]: 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]]])
 In [5]: #Import numpy package as np
         #Define a ndarray x2, whose shape is (3, 2, 2) and contains all 1's.
         #Define a ndarray x3, whose shape is (4,4) and contains 1's on diagonal
          and 0's elsewhere.
In [61]: x = np.eye(4)
         Χ
Out[61]: array([[1., 0., 0., 0.],
                [0., 1., 0., 0.],
                [0., 0., 1., 0.],
                [0., 0., 0., 1.]]
In [65]: np.random.seed(100)
         x=np.random.randint(0,1,24)
         x1=np.full(shape=(3,4,2),fill value=1)
         x1
Out[65]: 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]])
In [66]: x1 = np.random.rand(3,4,2)
         x1
Out[66]: array([[[0.54340494, 0.27836939],
                 [0.42451759, 0.84477613],
                 [0.00471886, 0.12156912],
                 [0.67074908, 0.82585276]],
                [[0.13670659, 0.57509333],
                 [0.89132195, 0.20920212],
                 [0.18532822, 0.10837689],
                 [0.21969749, 0.97862378]],
                [[0.81168315, 0.17194101],
                 [0.81622475, 0.27407375],
                 [0.43170418, 0.94002982],
                 [0.81764938, 0.33611195]])
 In [4]: #Define a random 3-D array x4 of shape (3, 4, 2) and of numbers between
          0 and 1.
         ##Simulate a random normal distribution of 20 elements, whose mean is 5
          and standard deviation 2.5 . Capture the result in x5.
 In [3]: import numpy as np
         np.random.seed(100)
         x5 = 5 + 10*np.random.randn(3,4,2)
         x5
 Out[3]: array([[[-12.49765473,
                                  8.42680403],
```



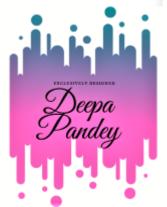
```
[ 16.53035803, 2.47563963],
                 [ 14.81320787, 10.14218841],
                 [ 7.21179669, -5.70043331]],
                [[ 3.10504169,
                                 7.550014441,
                 [ 0.41973014,
                                 9.35163488],
                 [ -0.8359505 , 13.16847072],
                 [ 11.72720806,
                                 3.95588857]],
                [[-0.31280377, 15.29732685],
                 [ 0.61864377, -6.18318246],
                 [ 21.18981661, 20.41605175],
                 [ 2.48120861, -3.42435738]]])
 In [6]: np.random.seed(100)
         x1 = np.arange(0,22,2)
         x1
 Out[6]: array([ 0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20])
 In [9]: x2 = np.linspace(10, 20, 30)
         x2
 Out[9]: array([10.
                           , 10.34482759, 10.68965517, 11.03448276, 11.37931034,
                11.72413793, 12.06896552, 12.4137931 , 12.75862069, 13.10344828,
                13.44827586, 13.79310345, 14.13793103, 14.48275862, 14.82758621,
                15.17241379, 15.51724138, 15.86206897, 16.20689655, 16.55172414,
                16.89655172, 17.24137931, 17.5862069 , 17.93103448, 18.27586207,
                18.62068966, 18.96551724, 19.31034483, 19.65517241, 20.
         ])
In [10]: import numpy as np
         x = np.arange(4).reshape(2,2)
         y = np.vsplit(x,2)
         print(y[0])
         [[0 1]]
```



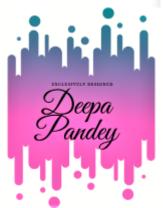
```
In [11]: import numpy as np
         x = np.arange(4).reshape(2,2)
         y = np.arange(4, 8).reshape(2,2)
         print(np.hstack((x,y)))
         [[0 1 4 5]
          [2 3 6 7]]
In [15]: import numpy as np
         x = np.arange(90).reshape(3, 15, 2)
         x.size[1]
         TypeError
                                                    Traceback (most recent call l
         ast)
         <ipython-input-15-1807c7a1fb8b> in <module>
               3 \times = np.arange(90).reshape(3, 15, 2)
         ----> 4 x.size[1]
         TypeError: 'int' object is not subscriptable
In [16]: import numpy as np
         x = np.arange(20).reshape(10, 10)
         x.shape
         ValueError
                                                    Traceback (most recent call l
         ast)
         <ipython-input-16-c1c2b803be39> in <module>
               1 import numpy as np
               2
```



```
---> 3 x = np.arange(20).reshape(10, 10)
               4 x.shape
         ValueError: cannot reshape array of size 20 into shape (10,10)
In [17]: #Create a ndarray x having first 20 natural numbers, using arange metho
         \#Determine the shape of x.
         #Change the shape of x to (2, 10) and assign it to new array y.
         #Split the array y horizontally in to two arrays.
In [22]: x = np.arange(20)
         x.shape
         y = x.reshape(2,10)
         y = np.hsplit(y,2)
Out[22]: [array([[ 0, 1, 2, 3, 4],
                 [10, 11, 12, 13, 14]]),
          array([[ 5, 6, 7, 8, 9],
                 [15, 16, 17, 18, 19]])]
In [23]: #Try it Out - Array Manipulation 2
         \#Now, Change the shape of x to (4, 5) and assign it to new array z.
         #Split the array z vertically in to two arrays.
In [24]: x = np.arange(20)
         z = x.reshape(4,5)
         res = np.vsplit(z,2)
         res
Out[24]: [array([[0, 1, 2, 3, 4],
                 [5, 6, 7, 8, 9]]),
```



```
array([[10, 11, 12, 13, 14],
                 [15, 16, 17, 18, 19]])]
In [25]: #Try it Out - Join Arrays
         #Create a 2-D array p, of shape (2, 2) with elements 3, 6, 9, 12.
         #Create a 2-D array q, of shape (2, 3) with elements 15, 18, 21, 24, 2
         7, 30.
         #Join the two arrays p and g horizontally.
In [33]: p = np.linspace(3, 12, 4)
         p = p.reshape(2,2)
         q = np.linspace(15,30,6)
         q = q.reshape(2,3)
         res = np.hstack((p,q))
         res
Out[33]: array([[ 3., 6., 15., 18., 21.],
                [ 9., 12., 24., 27., 30.]])
In [34]: import numpy as np
         x = np.arange(4).reshape(2,2)
         print(np.isfinite(x))
         [[ True True]
          [ True True]]
In [35]: import numpy as np
         x = np.arange(30).reshape(5,6)
         print(x.argmax(axis=1))
         [5 5 5 5 5]
In [36]: import numpy as np
```



```
x = np.array([[-2],
                         [2]])
          y = np.array([[-3, 3]])
          print(x + y)
          [[-5 1]
          \begin{bmatrix} -1 & 5 \end{bmatrix}
In [37]: import numpy as np
          print(np.repeat(3, 4))
          [3 3 3 3]
In [38]: import numpy as np
          x = np.arange(4).reshape(2,2)
          print(x.tolist())
          [[0, 1], [2, 3]]
In [39]: #Try it Out - Operations on Arrays 1
          #Create a 2-D array y of shape (2, 3), having numbers from 1 to 6.
          #Square each element of y.
          #Add 5 to each element of resulted array.
In [44]: y = np.linspace(1,6,6).reshape(2,3)
          y = y^{**}2
          y = y+5
Out[44]: array([[ 6., 9., 14.],
                 [21., 30., 41.]])
In [45]: \#Create\ a\ array\ x\ of\ shape\ (5,\ 6), having random integers between -30 a
          nd 30.
```



```
\#Determine the cumulative sum of x along axis 0.
         \#Determine the cumulative sum of x along axis 1.
In [50]: x = np.random.randint(-30,30,size=(5,6))
         p = x.sum(axis=0)
         q = x.sum(axis=1)
Out[50]: array([ 78, 10, -37, 14, -40])
In [51]: import numpy as np
         x = np.array([[1, 2], [3, 4], [5, 6]])
         print(x[[0, 1, 2], [0, 1, 1]])
         [1 4 6]
In [52]: import numpy as np
         x = np.arange(4)
         print(x.flatten())
         [0 1 2 3]
In [53]: import numpy as np
         x = np.arange(30).reshape(3,5,2)
         print(x[1,::2,1])
         [11 15 19]
In [54]: import numpy as np
         x = np.arange(30).reshape(3,5,2)
         print(x[1][::2][1])
```



```
[14 15]
In [55]: import numpy as np
         x = np.array([[0, 1], [1, 1], [2, 2]])
         print(x.sum(-1))
         [1 2 4]
In [56]: \#Create\ a\ array\ x\ of\ shape\ (6,\ 5),\ having\ first\ 30\ natural\ numbers.
         #Obtain elements of last row.
         #Obtain elements of middle column.
         #Obtain elements, overlapping first two rows and last three columns.
In [65]: x = np.arange(30).reshape(6,5)
         x[4]
         x[:,2]
Out[65]: array([ 2, 7, 12, 17, 22, 27])
In []: \#Create\ a\ array\ x\ of\ shape\ (2, 3, 5),\ having\ first\ 30\ natural\ numbers.
         #Create a boolean array b of shape (2,), having elements True, False.
         #Try the following expressions
         \#x[b]
         #x[b,:,1:3]
In [67]: x = np.arange(30).reshape(2, 3, 5)
         b = np.array([True,False])
         x[b]
         x[b,:,1:3]
```



```
Out[67]: array([[[ 1, 2],
                 [6, 7],
                 [11, 12]])
In [68]: import numpy as np
         print(np.array(([1, 2], (3,4))).shape)
         (2, 2)
In [69]: import numpy as np
         x = np.array([[0, 1], [1, 1], [2, 2]])
         y = x.sum(-1)
         print(x[y < 2, :])
         [[0 1]]
In [70]: import numpy as np
         x = np.arange(30).reshape(3,5,2)
         print(x[1][::2][1])
         [14 15]
In [71]: import numpy as np
         x = np.arange(30).reshape(5,6)
         print(x.argmax(axis=1))
         [5 5 5 5 5]
In [72]: import numpy as np
         x = np.arange(4)
         y = np.arange(4)
         print(x == y)
```

