lab-7

In [1]:

```
import numpy as np
a = np.array([12,23,34,56,67,78,89])
a = np.array([12,23,34,56,67,78,89])
c = np.array([[1,2,3],[4,5,6],[7,8,9]])
b = np.array([[1,2],[3,4]])
print(a)
```

```
[12 23 34 56 67 78 89]
```

In [2]:

```
print(b)
```

```
[[1 2]
 [3 4]]
```

In [3]:

```
print(c)
```

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

In [4]:

```
np.ones((3,4))
```

Out[4]:

```
array([[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]])
```

In [5]:

```
np.zeros((3,4))
```

Out[5]:

```
array([[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]])
```

In [8]:

```
np.empty((3,4))
```

Out[8]:

```
array([[0.83679635, 0.33878413, 0.88221426, 0.3027776 ],
       [0.56304291, 0.67912013, 0.89683732, 0.41678799],
       [0.51842204, 0.84503403, 0.65988784, 0.0743788 ]])
```

In [9]:

```
np.random.random((3,4))
```

Out[9]:

```
array([[0.28586235, 0.90463268, 0.59362499, 0.12041353],
       [0.35147005, 0.69713808, 0.48170364, 0.40645737],
       [0.62099458, 0.79859678, 0.98847289, 0.2534028 ]])
```

In [10]:

```
np.ones((3,4), dtype = np.int32)
```

Out[10]:

```
array([[1, 1, 1, 1],
       [1, 1, 1, 1],
       [1, 1, 1, 1]], dtype=int32)
```

In [11]:

```
np.full((2,2),7)
```

Out[11]:

```
array([[7, 7],
       [7, 7]])
```

In [12]:

```
np.full((4,2),24)
```

Out[12]:

```
array([[24, 24],
       [24, 24],
       [24, 24],
       [24, 24]])
```

In [13]:

```
np.eye(5)
```

Out[13]:

```
array([[1., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0.],
       [0., 0., 1., 0., 0.],
       [0., 0., 0., 1., 0.],
       [0., 0., 0., 0., 1.]])
```

In [14]:

```
np.identity(3)
```

Out[14]:

```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

In [15]:

```
np.diag(np.array([1,2,3,4]))
```

Out[15]:

```
array([[1, 0, 0, 0],
       [0, 2, 0, 0],
       [0, 0, 3, 0],
       [0, 0, 0, 4]])
```

In [16]:

```
np.diag(np.array([51,22,36,10,423]))
```

Out[16]:

```
array([[ 51,   0,   0,   0,   0],
       [  0,  22,   0,   0,   0],
       [  0,   0,  36,   0,   0],
       [  0,   0,   0,  10,   0],
       [  0,   0,   0,   0, 423]])
```

In [17]:

```
a=np.array([12,23,34,56,67,78,89])
print(a)
```

```
[12 23 34 56 67 78 89]
```

In [18]:

```
print(a.data)
```

```
<memory at 0x7f58286bda08>
```

In [19]:

```
print(a.dtype)
```

```
int64
```

In [21]:

```
print(a.shape)
```

```
(7,)
```

In [22]:

```
print(a.strides)
```

```
(8,)
```

In [24]:

```
print(a.ndim)
```

```
1
```

In [25]:

```
print(a.size)
```

```
7
```

In [26]:

```
print(a.itemsize)
```

```
8
```

In [27]:

```
print(len(a))
```

```
7
```

In [28]:

```
print(b)
print(len(b))
```

```
[[1 2]
 [3 4]]
2
```

In [29]:

```
print(b)
print(b.dtype)
b = b.astype(float)
print(b)
print(b.dtype)
```

```
[[1 2]
 [3 4]]
int64
[[1. 2.]
 [3. 4.]]
float64
```

In [30]:

```
np.arange(11)
```

Out[30]:

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

In [31]:

```
np.arange(26)
```

Out[31]:

```
array([ 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])
```

In [32]:

```
np.arange(10,21)
```

Out[32]:

```
array([10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20])
```

In [33]:

```
np.arange(15,26)
```

Out[33]:

```
array([15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25])
```

In [34]:

```
np.arange(10,25,5)
```

Out[34]:

```
array([10, 15, 20])
```

In [35]:

```
np.arange(1,25,3)
```

Out[35]:

```
array([ 1,  4,  7, 10, 13, 16, 19, 22])
```

In [36]:

```
np.arange(1,25,3.0)
```

Out[36]:

```
array([ 1.,  4.,  7., 10., 13., 16., 19., 22.])
```

In [37]:

```
np.arange(1,25,3, dtype=np.float32)
```

Out[37]:

```
array([ 1.,  4.,  7., 10., 13., 16., 19., 22.], dtype=float32)
```

In [38]:

```
print(np.arange(1,25,3, dtype=np.float32))
```

```
[ 1.  4.  7. 10. 13. 16. 19. 22.]
```

In [39]:

```
np.linspace(1,9,9)
```

Out[39]:

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

In [40]:

```
np.linspace(1,9,18)
```

Out[40]:

```
array([1.        , 1.47058824, 1.94117647, 2.41176471, 2.88235294,
       3.35294118, 3.82352941, 4.29411765, 4.76470588, 5.23529412,
       5.70588235, 6.17647059, 6.64705882, 7.11764706, 7.58823529,
       8.05882353, 8.52941176, 9.        ])
```

In [41]:

```
a=np.array([12,23,34,56,67,78,89])
print(a)
```

```
[12 23 34 56 67 78 89]
```

In [42]:

```
a[0]
```

Out[42]:

```
12
```

In [43]:

```
a[4]
```

Out[43]:

```
67
```

In [44]:

```
a[20]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-44-2ba6acdce15e> in <module>
----> 1 a[20]

IndexError: index 20 is out of bounds for axis 0 with size 7
```

In [45]:

```
a[-1]
```

Out[45]:

```
89
```

In [46]:

```
a[-7]
```

Out[46]:

```
12
```

In [47]:

```
a[-6]
```

Out[47]:

```
23
```

In [48]:

```
data=array([[11,22], [33,44], [55,66]])
print(data)
print(data[0,0])
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-48-c60fc73fcbb4> in <module>
----> 1 data=array([[11,22], [33,44], [55,66]])
      2 print(data)
      3 print(data[0,0])

NameError: name 'array' is not defined
```

In [51]:

```
from numpy import array
data=array([[11,22], [33,44], [55,66]])
print(data)
print(data[0,0])
```

```
[[11 22]
 [33 44]
 [55 66]]
11
```

In [52]:

```
print(data[0,])
```

```
[11 22]
```

In [54]:

```
print(data[0,...])
```

```
[11 22]
```

In [55]:

```
print(data[0,;])
```

```
  File "<ipython-input-55-7b8957819434>", line 1
    print(data[0,;])
                 ^
SyntaxError: invalid syntax
```

In [56]:

```
print(data[0,:])
```

```
[11 22]
```

In [57]:

```
print(data[,0])
```

```
  File "<ipython-input-57-5f07b93c5069>", line 1
    print(data[,0])
                ^
SyntaxError: invalid syntax
```

In [58]:

```
print(data[...,0])
```

```
[11 33 55]
```

In [59]:

```
print(data[:,0])
```

```
[11 33 55]
```

In [60]:

```
print(data[1:])
```

```
[[33 44]
 [55 66]]
```

In [61]:

```
print(a[:])
```

```
[12 23 34 56 67 78 89]
```

In [62]:

```
print(a[0:4])
```

```
[12 23 34 56]
```

In [63]:

```
print(a[-4:])
```

```
[56 67 78 89]
```

In [64]:

```
print(a[2:])
```

```
[34 56 67 78 89]
```

In [65]:

```
print(a)
print(a[0::2])
```

```
[12 23 34 56 67 78 89]
[12 34 67 89]
```

In [66]:

```
print(a)
print(a[1::2])
```

```
[12 23 34 56 67 78 89]
[23 56 78]
```

In [67]:

```
x=np.array([0,1,2,3,4,5,6,7,8,9])
print(x)
print(x[1:7:2])
```

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

In [68]:

```
print(x)
print(x[2:7:3])
```

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

In [69]:

```
print(x)
print("print elements from index 8 and 9 using positive indexing")
print(x[8:10])
print("print elements from index 8 and 9 using negitive indexing")
print(x[-2:10])
```

```
[0 1 2 3 4 5 6 7 8 9]
print elements from index 8 and 9 using positive indexing
[8 9]
print elements from index 8 and 9 using negitive indexing
[8 9]
```

In [71]:

```
print(x)
print(x[-3:3:-1])
```

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

In [72]:

```
from numpy import array
data = array([[11,22,33], [44,55,66], [77,88,99]])
x = data[:, :-1]
y = data[:, -1]
print("output of given array data is:")
print(data)
print("output of print(x)")
print(x)
print("output of print(y)")
print(y)
```

```
output of given array data is:
[[11 22 33]
 [44 55 66]
 [77 88 99]]
output of print(x)
[[11 22]
 [44 55]
 [77 88]]
output of print(y)
[33 66 99]
```

In [73]:

```
from numpy import array
data = array([11,22,33,44,55])
print(data.shape)
(5,)
from numpy import array
data=array([[11,22], [33,44], [55,66]])
print(data.shape)
(3, 2)
print('Rows: %d' % data.shape[0])
print('Cols: %d' % data.shape[1])
```

```
(5,)
(3, 2)
Rows: 3
Cols: 2
```

In [74]:

```
data=np.zeros((2,3,4))
print(data)
print(data.shape)
```

```
[[[0. 0. 0. 0.]
  [0. 0. 0. 0.]
  [0. 0. 0. 0.]]

 [[0. 0. 0. 0.]
  [0. 0. 0. 0.]
  [0. 0. 0. 0.]]]
(2, 3, 4)
```

In [75]:

```
data=np.zeros((2,3,4))
print("Given data is:")
print(data)
data.shape=(3,8)
print("Reshaped data is")
print(data)
```

```
Given data is:
[[[0. 0. 0. 0.]
  [0. 0. 0. 0.]
  [0. 0. 0. 0.]]

 [[0. 0. 0. 0.]
  [0. 0. 0. 0.]
  [0. 0. 0. 0.]]]
Reshaped data is
[[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 [76]:

```
data = np.random.random((2,3,4))
print("Given data is:")
print(data)
data.shape=(12,2)
print("Reshaped data is")
print(data)
```

```
Given data is:
[[[0.39656402 0.64705291 0.38823051 0.99021858]
  [0.59839615 0.97427149 0.62836392 0.84059906]
  [0.59490082 0.40595108 0.14528449 0.94052371]]

 [[0.35711425 0.39382471 0.82478243 0.34087704]
  [0.56723005 0.06079489 0.10858685 0.63352514]
  [0.17682252 0.4068875  0.17083745 0.83063852]]]
Reshaped data is
[[0.39656402 0.64705291]
 [0.38823051 0.99021858]
 [0.59839615 0.97427149]
 [0.62836392 0.84059906]
 [0.59490082 0.40595108]
 [0.14528449 0.94052371]
 [0.35711425 0.39382471]
 [0.82478243 0.34087704]
 [0.56723005 0.06079489]
 [0.10858685 0.63352514]
 [0.17682252 0.4068875 ]
 [0.17083745 0.83063852]]
```

In [77]:

```
data=np.ones((3,3,2), dtype=np.int)
print("Given data is:")
print(data)
data.shape=(1,18)
print("Reshaped data is")
print(data)
```

```
Given data is:
[[[1 1]
  [1 1]
  [1 1]]

 [[1 1]
  [1 1]
  [1 1]]

 [[1 1]
  [1 1]
  [1 1]]]
Reshaped data is
[[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]]
```

In [78]:

```
data=np.ones((2,3,5))
print("Given data is:")
print(data)data.shape=(12,2)
print("Reshaped data is")
print(data)
```

```
  File "<ipython-input-78-153c1789d62d>", line 3
    print(data)data.shape=(12,2)
                  ^
SyntaxError: invalid syntax
```

In [80]:

```
data=np.ones((2,3,5))
print("Given data is:")
print(data)
data.shape=(12,2)
print("Reshaped data is")
print(data)
```

```
Given data is:
[[[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.]]]

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-80-4f4aa58b47df> in <module>
      2 print("Given data is:")
      3 print(data)
----> 4 data.shape=(12,2)
      5 print("Reshaped data is")
      6 print(data)

ValueError: cannot reshape array of size 30 into shape (12,2)
```

In [81]:

```
a=np.arange(6).reshape((3,2))
print(a)
```

```
[[0 1]
 [2 3]
 [4 5]]
```

In [82]:

```
b=np.reshape(a, (2,3))
print(b)
```

```
[[0 1 2]
 [3 4 5]]
```

In [83]:

```
b=a.T
print(b)
```

```
[[0 2 4]
 [1 3 5]]
```

In [92]:

```
a=np.array([[1,2,3], [4,5,6]])
print("Given array is")
print(a)
b=np.reshape(a,6)
print("C-like index ordering")
print(b)
print("Fortran-like index ordering")
c=np.reshape(a,6, order='F')
print(c)
```

```
Given array is
[[1 2 3]
 [4 5 6]]
C-like index ordering
[1 2 3 4 5 6]
Fortran-like index ordering
[1 4 2 5 3 6]
```

In [93]:

```
import numpy as np
a=np.array([1,2,3,4])
b=np.array([5,6,7,8])
conc_h=np.concatenate((a,b), axis= 0)
print(conc_h)
```

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

In [96]:

```
a=np.array([1,2,3,4])
b=np.array([5,6,7,8])
conc_v=np.concatenate((a,b), axis=1)
print(conc_v)
```

```
---------------------------------------------------------------------------
AxisError                                 Traceback (most recent call last)
<ipython-input-96-021d5ae08d33> in <module>
      1 a=np.array([1,2,3,4])
      2 b=np.array([5,6,7,8])
----> 3 conc_v=np.concatenate((a,b), axis=1)
      4 print(conc_v)

<__array_function__ internals> in concatenate(*args, **kwargs)

AxisError: axis 1 is out of bounds for array of dimension 1
```

In [97]:

```
import numpy as np
a=np.array([[1,2],[3,4]])
b=np.array([[5,6],[7,8]])
conc_v=np.concatenate((a,b), axis=0)
print(conc_v)
```

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

In [98]:

```
a=np.array([[1,2],[3,4]])
b=np.array([[5,6],[7,8]])
conc_h=np.concatenate((a,b), axis=1)
print(conc_h)
```

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

In [99]:

```
import numpy as np
a=np.array([[1,2],[3,4]])
b=np.array([[5,6],[7,8]])
conc_n=np.concatenate((a,b), axis=None)
print(conc_n)
```

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

In [100]:

```
x=np.arange(9)
np.split(x,1)
[array([0, 1, 2, 3, 4, 5, 6, 7, 8])]
x=np.arange(9)
np.split(x,2)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
/srv/conda/envs/notebook/lib/python3.6/site-packages/numpy/lib/shape_base.py
in split(ary, indices_or_sections, axis)
    866     try:
--> 867         len(indices_or_sections)
    868     except TypeError:

TypeError: object of type 'int' has no len()

During handling of the above exception, another exception occurred:

ValueError                                Traceback (most recent call last)
<ipython-input-100-7633c2e1a27d> in <module>
      3 [array([0, 1, 2, 3, 4, 5, 6, 7, 8])]
      4 x=np.arange(9)
----> 5 np.split(x,2)

<__array_function__ internals> in split(*args, **kwargs)

/srv/conda/envs/notebook/lib/python3.6/site-packages/numpy/lib/shape_base.py
in split(ary, indices_or_sections, axis)
    871         if N % sections:
    872             raise ValueError(
--> 873                 'array split does not result in an equal division')
    874     return array_split(ary, indices_or_sections, axis)
    875

ValueError: array split does not result in an equal division
```

In [101]:

```
print('First array:')
print(a)
print('\n')
print('Horizontal splitting:')
np.hsplit(a,2)
```

```
First array:
[[1 2]
 [3 4]]


Horizontal splitting:
```

Out[101]:

```
[array([[1],
        [3]]),
 array([[2],
        [4]])]
```

In [102]:

```
a=np.arange(16).reshape(4,4)
print('First array:')
print(a)
print('\n')
print('Horizontal splitting:')
np.vsplit(a,2)
```

```
First array:
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]
 [12 13 14 15]]


Horizontal splitting:
```

Out[102]:

```
[array([[0, 1, 2, 3],
        [4, 5, 6, 7]]),
 array([[ 8,  9, 10, 11],
        [12, 13, 14, 15]])]
```

In [103]:

```
a=np.arange(16).reshape(4,4)
print('First array:')
print(a)
print('\n')
print('Horizontal splitting:')
np.vsplit(a,3)
```

```
First array:
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]
 [12 13 14 15]]


Horizontal splitting:

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
/srv/conda/envs/notebook/lib/python3.6/site-packages/numpy/lib/shape_base.py
in split(ary, indices_or_sections, axis)
    866     try:
--> 867         len(indices_or_sections)
    868     except TypeError:

TypeError: object of type 'int' has no len()

During handling of the above exception, another exception occurred:

ValueError                                Traceback (most recent call last)
<ipython-input-103-dd02c89db775> in <module>
      4 print('\n')
      5 print('Horizontal splitting:')
----> 6 np.vsplit(a,3)

<__array_function__ internals> in vsplit(*args, **kwargs)

/srv/conda/envs/notebook/lib/python3.6/site-packages/numpy/lib/shape_base.py
in vsplit(ary, indices_or_sections)
    989     if _nx.ndim(ary) < 2:
    990         raise ValueError('vsplit only works on arrays of 2 or more d
imensions')
--> 991     return split(ary, indices_or_sections, 0)
    992
    993

<__array_function__ internals> in split(*args, **kwargs)

/srv/conda/envs/notebook/lib/python3.6/site-packages/numpy/lib/shape_base.py
in split(ary, indices_or_sections, axis)
    871         if N % sections:
    872             raise ValueError(
--> 873                 'array split does not result in an equal division')
    874     return array_split(ary, indices_or_sections, axis)
    875

ValueError: array split does not result in an equal division
```

In [104]:

```
a=np.arange(16).reshape(4,4)
print('First array:')
print(a)
print('\n')
print('Horizontal splitting:')
np.vsplit(a,4)
```

```
First array:
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]
 [12 13 14 15]]


Horizontal splitting:
```

Out[104]:

```
[array([[0, 1, 2, 3]]),
 array([[4, 5, 6, 7]]),
 array([[ 8,  9, 10, 11]]),
 array([[12, 13, 14, 15]])]
```

In [106]:

```
a=np.arange(16).reshape(4,4)
print('First array:')
print(a)
print('\n')
print('Horizontal splitting:')
np.vsplit(a,[2,3])
```

```
First array:
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]
 [12 13 14 15]]


Horizontal splitting:
```

Out[106]:

```
[array([[0, 1, 2, 3],
        [4, 5, 6, 7]]),
 array([[ 8,  9, 10, 11]]),
 array([[12, 13, 14, 15]])]
```

In [107]:

```
a=np.array((1,2,3))
b=np.array((4,5,6))
np.hstack((a,b))
```

Out[107]:

```
array([1, 2, 3, 4, 5, 6])
```

In [108]:

```
a=np.array((1,2,3))
b=np.array((4,5,6))
np.vstack((a,b))
```

Out[108]:

```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [109]:

```
a=np.array([1,2,3])
b=np.array([4,5,6])
np.vstack((a,b))
```

Out[109]:

```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [110]:

```
a=np.array([1,2,3])
b=np.array([4,5,6])
np.hstack((a,b))
```

Out[110]:

```
array([1, 2, 3, 4, 5, 6])
```

In [111]:

```
a=np.array([23,87,34,90,32,334,76,12])
print("Given array is")
print(a)
print("\n")
print("sum of elements in the given array")
print(np.sum(a))
print("\n")
print("product of elements in the given array")
print(np.prod(a))
print("\n")
print("mean of elements in the given array")
print(np.mean(a))
print("\n")
print("standard deviation of elements in the given array")
print(np.std(a))
print("\n")
print("variance of elements in the given array")
print(np.var(a))
print("\n")
```

```
Given array is
[ 23  87  34  90  32 334  76  12]


sum of elements in the given array
688


product of elements in the given array
59684257935360


mean of elements in the given array
86.0


standard deviation of elements in the given array
97.90684347889069


variance of elements in the given array
9585.75
```

In [115]:

```
print("minimum element in the given array")
print(np.min(a)) # prints minimum element
print("\n")
print("maximum element in the given array")
print(np.max(a)) # prints maximum element
print("\n")
print("index of mimimum element in the given array")
print(np.argmin(a)) # prints index of mimimum element
print("\n")
print("index of maximum element in the given array")
print(np.argmax(a)) # prints index of maximum element
print("\n")
print("median of elements in the given array")
print(np.median(a)) # prints median of elements
print("\n")
```

```
minimum element in the given array
12


maximum element in the given array
334


index of mimimum element in the given array
7


index of maximum element in the given array
5


median of elements in the given array
55.0
```

In [116]:

```
print("rank based statistics of elements in the given array")
print("25th percentile")
print(np.percentile(a,25)) # prints 25th percentile
print("50th percentile (median)")
print(np.percentile(a,50)) # prints 50th percentile (median)
print("75th percentile")
print(np.percentile(a,75)) # prints 75th percentile
print("\n")
```

```
rank based statistics of elements in the given array
25th percentile
29.75
50th percentile (median)
55.0
75th percentile
87.75
```

In [117]:

```
a=np.array([23,87,34,90,32,334,76,12])
print("Given array is")
print(a)
print("\n")
print("Sum of the elements in the given array is")
print(a.sum())
print("\n")
print("Minimum element in the given array is")
print(a.min())
print("\n")
print("Maximum element in the given array is")
print(a.max())
print("\n")
```

```
Given array is
[ 23  87  34  90  32 334  76  12]


Sum of the elements in the given array is
688


Minimum element in the given array is
12


Maximum element in the given array is
334
```

In [118]:

```
a=np.array([23,87,34,90,32,334,76,12])
print("Given array is")
print(a)
print("\n")
# apply some condition
b=a>50
print("logical array is")
print(b)
print("\n")
print("evaluates whether any elements are true")
print(np.any(b)) # evaluates whether any elements are true
print("\n")
print("evaluates whether all elements are true")
print(np.all(b)) # evaluates whether any elements are true
print("\n")
```

```
Given array is
[ 23  87  34  90  32 334  76  12]


logical array is
[False  True False  True False  True  True False]


evaluates whether any elements are true
True


evaluates whether all elements are true
False
```

In [119]:

```
x=np.array([[1,1], [2,2]])
print(x)
```

```
[[1 1]
 [2 2]]
```

In [120]:

```
x.sum(axis=0)
array([3, 3])
x[:,1].sum()
```

Out[120]:

```
3
```

In [121]:

```
x.sum(axis=1)
array([2, 4])
x[0, :].sum()
```

Out[121]:

```
2
```

In [122]:

```
x[1, :].sum()
```

Out[122]:

```
4
```

In [123]:

```
x=np.arange(4)
print("x =", x)
print("x + 5 =", x+5)
print("x - 5 =", x-5)
print("x * 2 =", x*2)
print("x / 2 =", x/2)
print("x // 2 =", x//2) # floor divisionprint("-x =",-x)
print("x ** 2 =", x**2)
print("x%2 =", x%2)
```

```
x = [0 1 2 3]
x + 5 = [5 6 7 8]
x - 5 = [-5 -4 -3 -2]
x * 2 = [0 2 4 6]
x / 2 = [0.  0.5 1.  1.5]
x // 2 = [0 0 1 1]
x ** 2 = [0 1 4 9]
x%2 = [0 1 0 1]
```

In [124]:

```
a=np.array([1,1,1])
b=np.array([2,2,2])
print("a =", a)
print("\n")
print("b =", b)
print("\n")
print("a + b =", a+b)
print("\n")
print("a - b =", a-b)
print("\n")
print("a * b =", a*b)
print("\n")
print("a / b =", a/b)
print("\n")
print("a // b =", a//b)
print("\n")
print("-a =",-a)
print("\n")
print("a ** b =", a**b)
print("\n")
print("a%b =", a%b)
```

```
a = [1 1 1]


b = [2 2 2]


a + b = [3 3 3]


a - b = [-1 -1 -1]


a * b = [2 2 2]


a / b = [0.5 0.5 0.5]


a // b = [0 0 0]


-a = [-1 -1 -1]


a ** b = [1 1 1]


a%b = [1 1 1]
```

In [125]:

```
a=np.array([1,1,1])
b=np.array([2,2,2,2])
print("a =", a)
print("\n")
print("b =", b)
print("\n")
print("a + b =", a+b)
print("\n")
print("a - b =", a-b)
print("\n")
print("a * b =", a*b)
print("\n")
print("a / b =", a/b)
print("\n")
print("a // b =", a//b)
print("\n")
print("-a =",-a)
print("\n")
print("a ** b =", a**b)
print("\n")
print("a%b =", a%b)
```

```
a = [1 1 1]


b = [2 2 2 2]



---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-125-56e708825b7d> in <module>
      5 print("b =", b)
      6 print("\n")
----> 7 print("a + b =", a+b)
      8 print("\n")
      9 print("a - b =", a-b)

ValueError: operands could not be broadcast together with shapes (3,) (4,)
```

In [126]:

```
x=np.array([-2,-1,0,1,2])
print(x)
abs(x)
```

```
[-2 -1  0  1  2]
```

Out[126]:

```
array([2, 1, 0, 1, 2])
```

In [127]:

```
print(x)
np.abs(x)
```

```
[-2 -1  0  1  2]
```

Out[127]:

```
array([2, 1, 0, 1, 2])
```

In [128]:

```
print(x)
np.absolute(x)
```

```
[-2 -1  0  1  2]
```

Out[128]:

```
array([2, 1, 0, 1, 2])
```

In [129]:

```
a=np.array([[-1,2,-3],[4,5,6]])
np.abs(a)
```

Out[129]:

```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [130]:

```
theta=np.linspace(0, np.pi,3)
print("theta=", theta)
print("sin(theta) =", np.sin(theta))
print("cos(theta) =", np.cos(theta))
print("tan(theta) =", np.tan(theta))
```

```
theta= [0.         1.57079633 3.14159265]
sin(theta) = [0.0000000e+00 1.0000000e+00 1.2246468e-16]
cos(theta) = [ 1.000000e+00  6.123234e-17 -1.000000e+00]
tan(theta) = [ 0.00000000e+00  1.63312394e+16 -1.22464680e-16]
```

In [131]:

```
x=[-1,0,1]
print("x=", x)
print("arcsin(x) =", np.arcsin(x))
print("arccos(x) =", np.arccos(x))
print("arctan(x) =", np.arctan(x))
```

```
x= [-1, 0, 1]
arcsin(x) = [-1.57079633  0.          1.57079633]
arccos(x) = [3.14159265 1.57079633 0.        ]
arctan(x) = [-0.78539816  0.          0.78539816]
```

In [133]:

```
x=[1,2,3]
print("x =", x)
print("e^x =", np.exp(x))
print("2^x =", np.exp2(x))
print("3^x =", np.power(3, x))
```

```
x = [1, 2, 3]
e^x = [ 2.71828183  7.3890561  20.08553692]
2^x = [2. 4. 8.]
3^x = [ 3  9 27]
```

In [134]:

```
x=[1,2,4,10]
print("x=", x)
print("ln(x) =", np.log(x))
print("log2(x) =", np.log2(x))
print("log10(x) =", np.log10(x))
```

```
x= [1, 2, 4, 10]
ln(x) = [0.         0.69314718 1.38629436 2.30258509]
log2(x) = [0.         1.         2.         3.32192809]
log10(x) = [0.         0.30103    0.60205999 1.        ]
```

In [137]:

```
name=['Alice','Bob','Cathy','Doug']
age=[25,45,37,19]
weight=[55.0,85.5,68.0,61.5]
data=np.zeros(4, dtype={'names':('name','age','weight'),'formats':
('U10','i4','f8')})
print(data.dtype)
```

```
[('name', '<U10'), ('age', '<i4'), ('weight', '<f8')]
```

In [138]:

```
data['name']=name
data['age']=age
data['weight']=weight
print(data)
```

```
[('Alice', 25, 55. ) ('Bob', 45, 85.5) ('Cathy', 37, 68. )
 ('Doug', 19, 61.5)]
```

In [139]:

```
data['name']
```

Out[139]:

```
array(['Alice', 'Bob', 'Cathy', 'Doug'], dtype='<U10')
```

In [140]:

```
data[0]
```

Out[140]:

```
('Alice', 25, 55.)
```

In [141]:

```
data[-1]['name']
```

Out[141]:

```
'Doug'
```

In [142]:

```
data[-1]['name']
```

Out[142]:

```
'Doug'
```

In [143]:

```
data[data['age']<30]['name']
```

Out[143]:

```
array(['Alice', 'Doug'], dtype='<U10')
```