Numpy

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

creating arrays using list and options

In [7]:
```
mylist=(1,2,3)
x=np.array(mylist)
x
```

Out[7]:
```
array([1, 2, 3])
```

In [8]:
```
y=np.array([1,2,3]) #1-D array
y
```

Out[8]:
```
array([1, 2, 3])
```

In [9]:
```
y = np.array([[1,2,3],[4,5,6]]) # 2-D array
y
```

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

In [11]:
```
y.shape  #shows rows and columns of 2-D array (rows,cols)
```

Out[11]:
```
(2, 3)
```

In [17]:
```
n = np.arange(0,30,2)  # creates array starting fro 0-30 with interval of 2
n
```

Out[17]:
```
array([ 0,  2,  4,  6,  8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28])
```

In [22]:
```
n = n.reshape(3,5) # creates array of 3 rows and 5 columns
n
```

Out[22]:
```
array([[ 0,  2,  4,  6,  8],
       [10, 12, 14, 16, 18],
       [20, 22, 24, 26, 28]])
```

In [28]:
```
n = np.linspace(0,4,9) #it creates array from 0  to 4 with 9 interval values
n
```

Out[28]:
```
array([0. , 0.5, 1. , 1.5, 2. , 2.5, 3. , 3.5, 4. ])
```

In [29]:
```
n = np.linspace(0,8,9) #it creates array from 0  to 8 with 9 interval values
n
```

Out[29]:
```
array([0., 1., 2., 3., 4., 5., 6., 7., 8.])
```

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

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

In [39]:
```
np.ones((3,3))  #creates array with all values as 1
```

Out[39]:
```
array([[1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.]])
```

In [40]:
```
np.zeros((2,3)) #creates array with all values as 0
```

Out[40]:
```
array([[0., 0., 0.],
       [0., 0., 0.]])
```

In [42]:
```
np.eye(3) #creates array with diagonal values as 1
```

Out[42]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

In [45]:
```
np.diag(y)
```

Out[45]:
```
array([1, 5])
```

In [46]:
```
np.array([1,2,3]*3)  #repeats set values
```

Out[46]:
```
array([1, 2, 3, 1, 2, 3, 1, 2, 3])
```

In [47]:
```
np.repeat([1,2,3],3) #repeats values
```

Out[47]:
```
array([1, 1, 1, 2, 2, 2, 3, 3, 3])
```

In [48]:
```
p=np.ones([2,3],int)
p
```

Out[48]:
```
array([[1, 1, 1],
       [1, 1, 1]])
```

In [49]:
```
np.vstack([p,2*p])
```

Out[49]:
```
array([[1, 1, 1],
       [1, 1, 1],
       [2, 2, 2],
       [2, 2, 2]])
```

In [50]:
```
np.hstack([p,2*p])
```

Out[50]:
```
array([[1, 1, 1, 2, 2, 2],
       [1, 1, 1, 2, 2, 2]])
```

Numpy arrays operations

In [51]:
```
x + y
```

Out[51]:
```
array([[2, 4, 6],
       [5, 7, 9]])
```

In [52]:
```
x
```

Out[52]:
```
array([1, 2, 3])
```

In [53]:
```
y
```

Out[53]:
```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [55]:
```
x**y
```

Out[55]:
```
array([[  1,   4,  27],
       [  1,  32, 729]], dtype=int32)
```

In [58]:
```
y.T   #transpose of y
```

Out[58]:
```
array([[1, 4],
       [2, 5],
       [3, 6]])
```

In [59]:
```
y.shape
```

Out[59]:
```
(2, 3)
```

In [60]:
```
y.T
```

Out[60]:
```
array([[1, 4],
       [2, 5],
       [3, 6]])
```

In [61]:
```
y.shape
```

Out[61]:
```
(2, 3)
```

In [63]:
```
y.dtype  #shows type of data in matrix
```

Out[63]:
```
dtype('int32')
```

In [65]:
```
y=y.astype('f')
y.dtype
```

Out[65]:
```
dtype('float32')
```

In [66]:
```
a =np.array([-4,-2,1,3,5])
a
```

Out[66]:
```
array([-4, -2,  1,  3,  5])
```

In [67]:
```
a.max()
```

Out[67]:
```
5
```

In [68]:
```
a.min()
```

Out[68]:
```
-4
```

In [69]:
```
a.mean()
```

Out[69]:
```
0.6
```

In [70]:
```
a.sum()
```

Out[70]:
```
3
```

In [71]:
```
a.argmax()
```

Out[71]:
```
4
```

In [72]:
```
a.argmin()
```

Out[72]:
```
0
```

Indexing and Slicing

In [74]:
```
s =np.arange(13)**2  #it creates array square of 1-13 numbers
s
```

Out[74]:
```
array([  0,   1,   4,   9,  16,  25,  36,  49,  64,  81, 100, 121, 144],
      dtype=int32)
```

In [75]:
```
s[0]
```

Out[75]:
```
0
```

In [76]:
```
s[4]
```

Out[76]:
```
16
```

In [77]:
```
s[0:3]
```

Out[77]:
```
array([0, 1, 4], dtype=int32)
```

In [78]:
```
s[3:7]
```

Out[78]:
```
array([ 9, 16, 25, 36], dtype=int32)
```

In [79]:
```
s[-4:]
```

Out[79]:
```
array([ 81, 100, 121, 144], dtype=int32)
```

In [80]:
```
s[-4:-2]
```

Out[80]:
```
array([ 81, 100], dtype=int32)
```

In [81]:
```
s[-4::-2]
```

Out[81]:
```
array([81, 49, 25,  9,  1], dtype=int32)
```

In [84]:
```
r =np.arange(36)
r.resize((6,6))
r
```

Out[84]:
```
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, 26, 27, 28, 29],
       [30, 31, 32, 33, 34, 35]])
```

In [85]:
```
r[3,3] #value at[3,3]
```

Out[85]:
```
21
```

In [86]:
```
r[3,3:6]  #3rd row 3-6 col values
```

Out[86]:
```
array([21, 22, 23])
```

In [87]:
```
r[:-2,:-1]
```

Out[87]:
```
array([[ 0,  1,  2,  3,  4],
       [ 6,  7,  8,  9, 10],
       [12, 13, 14, 15, 16],
       [18, 19, 20, 21, 22]])
```

In [88]:
```
r[r<20]
```

Out[88]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19])
```

In [89]:
```
r1=r[:2,:2]   #copy from r to r1
r1
```

Out[89]:
```
array([[0, 1],
       [6, 7]])
```

In [91]:
```
r1[:]=0  #assign all values as 0
r1
```

Out[91]:
```
array([[0, 0],
       [0, 0]])
```

In [92]:
```
r  #values also get change in original array so take care here
```

Out[92]:
```
array([[ 0,  0,  2,  3,  4,  5],
       [ 0,  0,  8,  9, 10, 11],
       [12, 13, 14, 15, 16, 17],
       [18, 19, 20, 21, 22, 23],
       [24, 25, 26, 27, 28, 29],
       [30, 31, 32, 33, 34, 35]])
```

In [93]:
```
rcopy=r.copy()   #when we use copy() orignal values of array never get changed
rcopy
```

Out[93]:
```
array([[ 0,  0,  2,  3,  4,  5],
       [ 0,  0,  8,  9, 10, 11],
       [12, 13, 14, 15, 16, 17],
       [18, 19, 20, 21, 22, 23],
       [24, 25, 26, 27, 28, 29],
       [30, 31, 32, 33, 34, 35]])
```

In [95]:
```
rcopy[:]=0
rcopy
```

Out[95]:
```
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]])
```

In [96]:
```
r
```

Out[96]:
```
array([[ 0,  0,  2,  3,  4,  5],
       [ 0,  0,  8,  9, 10, 11],
       [12, 13, 14, 15, 16, 17],
       [18, 19, 20, 21, 22, 23],
       [24, 25, 26, 27, 28, 29],
       [30, 31, 32, 33, 34, 35]])
```

In [97]:
```
old = np.array([[1, 1, 1],
                [1, 1, 1]])

new = old.copy()
new[:, 0] = 0

print(old)
```

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

Iterating over Arrays

In [102]:
```
temp =np.random.randint(0,10,(4,3))  #create 2-D array from 0-10 numbers having 4 rows and 3 cols
temp
```

Out[102]:
```
array([[5, 1, 4],
       [3, 0, 9],
       [2, 3, 6],
       [7, 8, 4]])
```

In [103]:
```
for row in temp:   #iterate array
    print(row)
```

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

In [104]:
```
for row in range(len(temp)):
    print(temp[row])
```

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

In [105]:
```
for i, row in enumerate(temp):  #row wise printing
    print('row',i,'is',row)
```

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

In [107]:

```
Requirement already satisfied: nbconvert in c:\users\jayvant\anaconda3\lib\site-packages (5.6.1)
Requirement already satisfied: jupyter-core in c:\users\jayvant\anaconda3\lib\site-packages (from nbconvert) (4.6.1)
Requirement already satisfied: mistune<2,>=0.8.1 in c:\users\jayvant\anaconda3\lib\site-packages (from nbconvert) (0.8.4)
Requirement already satisfied: traitlets>=4.2 in c:\users\jayvant\anaconda3\lib\site-packages (from nbconvert) (4.3.3)
Requirement already satisfied: pygments in c:\users\jayvant\anaconda3\lib\site-packages (from nbconvert) (2.5.2)
Requirement already satisfied: nbformat>=4.4 in c:\users\jayvant\anaconda3\lib\site-packages (from nbconvert) (5.0.4)
Requirement already satisfied: entrypoints>=0.2.2 in c:\users\jayvant\anaconda3\lib\site-packages (from nbconvert) (0.3)
Requirement already satisfied: pandocfilters>=1.4.1 in c:\users\jayvant\anaconda3\lib\site-packages (from nbconvert) (1.
4.2)
Requirement already satisfied: testpath in c:\users\jayvant\anaconda3\lib\site-packages (from nbconvert) (0.4.4)
Requirement already satisfied: jinja2>=2.4 in c:\users\jayvant\anaconda3\lib\site-packages (from nbconvert) (2.11.1)
Requirement already satisfied: defusedxml in c:\users\jayvant\anaconda3\lib\site-packages (from nbconvert) (0.6.0)
Requirement already satisfied: bleach in c:\users\jayvant\anaconda3\lib\site-packages (from nbconvert) (3.1.0)
Requirement already satisfied: pywin32>=1.0; sys_platform == "win32" in c:\users\jayvant\anaconda3\lib\site-packages (fro
m jupyter-core->nbconvert) (227)
Requirement already satisfied: ipython-genutils in c:\users\jayvant\anaconda3\lib\site-packages (from traitlets>=4.2->nbc
onvert) (0.2.0)
Requirement already satisfied: decorator in c:\users\jayvant\anaconda3\lib\site-packages (from traitlets>=4.2->nbconvert)
(4.4.1)
Requirement already satisfied: six in c:\users\jayvant\anaconda3\lib\site-packages (from traitlets>=4.2->nbconvert) (1.1
4.0)
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in c:\users\jayvant\anaconda3\lib\site-packages (from nbformat>=4.
4->nbconvert) (3.2.0)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\jayvant\anaconda3\lib\site-packages (from jinja2>=2.4->nbconv
ert) (1.1.1)
Requirement already satisfied: webencodings in c:\users\jayvant\anaconda3\lib\site-packages (from bleach->nbconvert) (0.
5.1)
Requirement already satisfied: attrs>=17.4.0 in c:\users\jayvant\anaconda3\lib\site-packages (from jsonschema!=2.5.0,>=2.
4->nbformat>=4.4->nbconvert) (19.3.0)
Requirement already satisfied: setuptools in c:\users\jayvant\anaconda3\lib\site-packages (from jsonschema!=2.5.0,>=2.4->
nbformat>=4.4->nbconvert) (45.2.0.post20200210)
Requirement already satisfied: importlib-metadata; python_version < "3.8" in c:\users\jayvant\anaconda3\lib\site-packages
(from jsonschema!=2.5.0,>=2.4->nbformat>=4.4->nbconvert) (1.5.0)
Requirement already satisfied: pyrsistent>=0.14.0 in c:\users\jayvant\anaconda3\lib\site-packages (from jsonschema!=2.5.
0,>=2.4->nbformat>=4.4->nbconvert) (0.15.7)
Requirement already satisfied: zipp>=0.5 in c:\users\jayvant\anaconda3\lib\site-packages (from importlib-metadata; python
_version < "3.8"->jsonschema!=2.5.0,>=2.4->nbformat>=4.4->nbconvert) (2.2.0)
Note: you may need to restart the kernel to use updated packages.
```

In [ ]: