Aditya_numpy

February 22, 2024

[1]:
```
print("Hello World")
```

```
Hello World
```

[2]:
```
import numpy as np
```

[3]:
```
a=np.array([1,2,3,4])
print(a)
```

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

[4]:
```
print("Array a : ",a)
print(f"Array a : {a}")
print("Array a : {}".format(a))
```

```
Array a :  [1 2 3 4]
Array a : [1 2 3 4]
Array a : [1 2 3 4]
```

[5]:
```
#dimesion of an array
print(f"ndim a : {a.ndim}")
```

```
ndim a : 1
```

[6]:
```
#size of an element of an array
print(f"size a : {a.size}")
```

```
size a : 4
```

[7]:
```
#shape of an array
print(f"shape a : {a.shape}")
```

```
shape a : (4,)
```

[8]:
```
#data type of an array
print(f"dtype a : {a.dtype}")
```

```
dtype a : int32
```

[9]:
```
#itemsize of an array
print(f"itemsize a : {a.itemsize}")
```

```
itemsize a : 4
```

[10]:
```
#two dimesional array
a=np.array([[1,2,3,4],[5,6,7,8]])
print(a)
```

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

[11]:
```
#adding an array
b=np.array([1,2,3,4])
c=np.add(a,b)
print(c)
```

```
[[ 2  4  6  8]
 [ 6  8 10 12]]
```

[12]:
```
#operating on an array with add, subtract, multiply, divide, etc...
f=np.array([[1,2,3,4],[5,6,7,8]])
c=np.add(a,b)
d=np.subtract(a,b)
e=np.multiply(a,b)
g=np.divide(a,b)
h=np.power(a,b)
i=np.mod(a,b)
j=np.remainder(a,b)
k=np.absolute(a)
print(f)
print(c)
print(d)
print(e)
print(g)
print(h)
print(i)
print(j)
print(k)
```

```
[[1 2 3 4]
 [5 6 7 8]]
[[ 2  4  6  8]
 [ 6  8 10 12]]
[[0 0 0 0]
 [4 4 4 4]]
[[ 1  4  9 16]
 [ 5 12 21 32]]
[[1.         1.         1.         1.        ]
 [5.         3.         2.33333333 2.        ]]
[[   1    4   27  256]
 [   5   36  343 4096]]
```

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

[13]:
```
#srting in an array
y=np.array(['Aditya','Mili','Jagrat','Parth','Krishna','Janvi','Dax'])
print(y)
```

```
['Aditya' 'Mili' 'Jagrat' 'Parth' 'Krishna' 'Janvi' 'Dax']
```

[14]:
```
#setting zeros in an array
z=np.zeros((2,3))
print(z)
```

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

[15]:
```
#Setting one in an array
one=np.ones((3,3))
print(one)
```

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

[16]:
```
#empty
empty=np.empty((2,2))
print(empty)
```

```
[[2.12199579e-314 1.06736388e-311]
 [2.23317672e-321 1.06736388e-311]]
```

[17]:
```
#arranging array
ar=np.arange(1,5,0.5)
print(ar)
```

```
[1.  1.5 2.  2.5 3.  3.5 4.  4.5]
```

[18]:
```
#linespace
l=np.linspace(1,5,10)
print(l)
```

```
[1.         1.44444444 1.88888889 2.33333333 2.77777778 3.22222222
 3.66666667 4.11111111 4.55555556 5.        ]
```

[36]:
```
r=np.random.random((2,3))     #random number printing in an array
print(r)
```

```
[[0.24323303 0.4346347  0.4728729 ]
 [0.26613219 0.31650001 0.02828841]]
```

[20]:
```
a=np.array([[2,3,4],[6,7,8]])
print(a)
sliced_arr=a[:2,:2]
print(f"sliced_arr : {sliced_arr}")     #Slicing an array
```

```
[[2 3 4]
 [6 7 8]]
sliced_arr : [[2 3]
 [6 7]]
```

[21]:
```
indexed_arr=a[[1,0],[0,1]]
print(indexed_arr)     #indexing an array
```

```
[6 3]
```

[22]:
```
print(a.sum())     #sum of an array
```

```
30
```

[23]:
```
a1=np.array([25,45,56,81])
print(np.sqrt(a1))     #sqrt of an array
```

```
[5.         6.70820393 7.48331477 9.        ]
```

[24]:
```
b=a+5
print(b)     #adding value to an array
```

```
[[ 7  8  9]
 [11 12 13]]
```

[25]:
```
m1=np.array([[1,2,3],[4,5,6],[7,8,9]])
m2=np.array([[4,5,6],[7,8,9],[4,7,9]])
m3=np.dot(m1,m2)     #multiplying an array using dot
print(m3)
```

```
[[ 30  42  51]
 [ 75 102 123]
 [120 162 195]]
```

[26]:
```
bool=np.array([[True,True,False],[True,False,False]])     #OR Operator used in␣
 ↪numpy as any()
print("any() with axis none ",np.any(bool))
print("any() with axis = 0 ",np.any(bool,axis=0))
print("any() with axis = 1 ",np.any(bool,axis=1))
```

```
any() with axis none  True
any() with axis = 0  [ True  True False]
any() with axis = 1  [ True  True]
```

[27]:
```
bool=np.array([[True,True,False],[True,False,False]])     #And Operator used in
 ↪numpy as all()
print("all() with axis none ",np.all(bool))
print("all() with axis = 0 ",np.all(bool,axis=0))
print("all() with axis = 1 ",np.all(bool,axis=1))
```

```
all() with axis none  False
all() with axis = 0  [ True False False]
all() with axis = 1  [False False]
```

[28]:
```
a1=np.arange(8)     #giving a range upto 8 elements
print(a1)
re=a1.reshape(2,4)     #Reshaping an element of an array into 2rows & 4columns
print(re)
```

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

[29]:
```
print(m1)     #printing m1
print(m2)     #printing m2
print(m3)     #printing m3
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
[[4 5 6]
 [7 8 9]
 [4 7 9]]
[[ 30  42  51]
 [ 75 102 123]
 [120 162 195]]
```

[30]:
```
v=np.vstack((m1,m2,m3))     #printing Vertical stack
print(v)
```

```
[[  1   2   3]
 [  4   5   6]
 [  7   8   9]
 [  4   5   6]
 [  7   8   9]
 [  4   7   9]
 [ 30  42  51]
 [ 75 102 123]
 [120 162 195]]
```

[31]:
```
h=np.hstack((m1,m2,m3))     #Printing Horizontal Stack
print(h)
```

```
[[  1   2   3   4   5   6  30  42  51]
```

```
 [  4   5   6   7   8   9  75 102 123]
 [  7   8   9   4   7   9 120 162 195]]
```

[32]:
```
hsplit=np.hsplit(h,3)      #splitting horizontally
print(hsplit)
```

```
[array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]]), array([[4, 5, 6],
       [7, 8, 9],
       [4, 7, 9]]), array([[ 30,  42,  51],
       [ 75, 102, 123],
       [120, 162, 195]])]
```

[33]:
```
vsplit=np.vsplit(h,3)      #Splitting Vertically
print(vsplit)
```

```
[array([[ 1,  2,  3,  4,  5,  6, 30, 42, 51]]), array([[  4,   5,   6,   7,   8,
9,  75, 102, 123]]), array([[  7,   8,   9,   4,   7,   9, 120, 162, 195]])]
```

[34]:
```
arr=np.array([23,34,45,67])
print("sqrt : ",np.sqrt(arr)) #Return the Square root of each element
print("exp  : ",np.exp(arr))  #Return the Exponentials of each element
print("sin  : ",np.sin(arr))  #Return the Sin of each element
print("cos  : ",np.cos(arr))  #Return the Cosine of each element
print("log  : ",np.log(arr))  #Return the Logarithm of each element
print("sum  : ",np.sum(arr))  #Return the Sum of each element
print("std  : ",np.std(arr))  #Return the Standard Deviation of each element
```

```
sqrt :  [4.79583152 5.83095189 6.70820393 8.18535277]
exp  :  [9.74480345e+09 5.83461743e+14 3.49342711e+19 1.25236317e+29]
sin  :  [-0.8462204   0.52908269  0.85090352 -0.85551998]
cos  :  [-0.53283302 -0.84857027  0.52532199 -0.5177698 ]
log  :  [3.13549422 3.52636052 3.80666249 4.20469262]
sum  :  169
std  :  16.269219403523945
```

[35]:
```
print("Random : {}\n".format(np.random.random(20)))      #20 Random numbers
print("Rand : {}\n".format(np.random.rand(3,4)))      #Random numbers in an␣
 ↪array of 3rows and 4columns
print("Randint : {}\n".format(np.random.randint(0,100,20)))      #printing 20␣
 ↪random numbers between 0 to 100
print("Permutation : {}\n".format(np.random.permutation(np.arange(20))))     ␣
 ↪#printing random numbers permutated
```

```
Random : [0.07466949 0.05170022 0.52509163 0.37794023 0.76749048 0.71017467
 0.22857409 0.77986629 0.61086049 0.25171391 0.93062862 0.13494146
 0.37370088 0.80888101 0.61489872 0.8506399  0.34814723 0.26955659
 0.06456865 0.16710519]
```

```
Rand : [[0.86829916 0.30781247 0.31748311 0.04467527]
 [0.83818517 0.69268568 0.4763982  0.32936644]
 [0.21146011 0.28666236 0.98763698 0.30397651]]

Randint : [45 65 20  4 30 60 27 85 76 48 24 15 71 41 49 41 90 12  5 23]

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