uses of numpy

ARRAY CREATION FN

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

creating an array

In [2]:
```
a=np.array([1,2,3,4,5])
print('Array is',a)
```

```
Array is [1 2 3 4 5]
```

In [3]:
```
b=np.arange(0,10,2)
print('array b is:',b)
```

```
array b is: [0 2 4 6 8]
```

In [6]:
```
c=np.ones((3,2))
print('ones array :\n',c)
```

```
ones array :
 [[1. 1.]
 [1. 1.]
 [1. 1.]]
```

In [9]:
```
d=np.zeros((3,2))
print('zeros array :\n',d)
```

```
zeros array :
 [[0. 0.]
 [0. 0.]
 [0. 0.]]
```

In [10]:
```
c=np.ones((3,2),dtype=int)
print(c)
```

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

In [11]:
```
d=np.zeros((3,2),dtype=int)
print(d)
```

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

ARRAY MANIPULATIOM FUNCTIONS

In [17]:
```
a1=np.array([1,2,3,4,5,6])
reshaped_arr=np.reshape(a1,(1,6))
print(a1)
```

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

In [19]:
```
reshaped_arr=np.reshape(a1,(6,1))
print(reshaped_arr)
```

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

In [20]:
```
b = np.array([[1, 2], [3, 4], [5, 6], [7, 8]])
flattened=np.ravel(b)
print('flattened array:\n',flattened)
```

```
flattened array:
 [1 2 3 4 5 6 7 8]
```

In [21]:
```
c = np.array([[1, 2], [3, 4]])
tranpose=np.transpose(c)
print('transposed array:\n',c)
```

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

In [22]:
```
a
```

Out[22]:
```
array([1, 2, 3, 4, 5])
```

In [23]:
```
b
```

Out[23]:
```
array([[1, 2],
       [3, 4],
       [5, 6],
       [7, 8]])
```

In [25]:
```
a=np.hstack(b)
print('stacked array horizontal:\n',a)
```

```
stacked array horizontal:
 [1 2 3 4 5 6 7 8]
```

In [26]:
```
b=np.hstack(a)
print('stacked array horizontal:\n',a)
```

```
stacked array horizontal:
 [1 2 3 4 5 6 7 8]
```

In [27]:
```
a=np.vstack(b)
print('stacked array vertical:\n',a)
```

```
stacked array vertical:
 [[1]
 [2]
 [3]
 [4]
 [5]
 [6]
 [7]
 [8]]
```

In [28]:
```
b=np.vstack(a)
print('stacked array vertical:\n',a)
```

```
stacked array vertical:
 [[1]
 [2]
 [3]
 [4]
 [5]
 [6]
 [7]
 [8]]
```

ADDING TWO ARRAYS

In [32]:
```
a = np.array([1, 2, 3, 4])
add=a+2
print('result of sum is:\n',add)
```

```
result of sum is:
 [3 4 5 6]
```

In [33]:
```
squared=np.power(a,2)
print(squared)
```

```
[ 1  4  9 16]
```

In [34]:
```
np.sqrt(squared)
```

Out[34]:
```
array([1., 2., 3., 4.])
```

RANDOM SAMPLING FUNCTIONS

In [35]:
```
s = np.random.rand(3)
print("Random values:", s)
```

```
Random values: [0.76518476 0.70484477 0.38219813]
```

In [37]:
```
np.random.seed(0)
s = np.random.rand(3)
print("Random values:", s)
```

```
Random values: [0.5488135  0.71518937 0.60276338]
```

In [38]:
```
e = np.random.randint(0, 10, size=5)

print("Random integers:", e)
```

```
Random integers: [3 7 9 3 5]
```

In [39]:
```
np.random.seed(0)
e = np.random.randint(0, 10, size=5)
print("Random integers:", e)
```

```
Random integers: [5 0 3 3 7]
```

BOOLEAN AND LOGIC OPERATIONS

In [40]:
```
test=np.array([True,False,True])
t= np.all(test)
print('are all elements true:',t)
```

```
are all elements true: False
```

In [43]:
```
test1=np.array([True,True,True])
t= np.all(test1)
print('are all elements true:',t)
```

```
are all elements true: True
```

In [42]:
```
test=np.array([True,False,True])
t= np.any(test)
print('are all elements true:',t)
```

```
are all elements true: True
```

SET OPERATIONS

In [45]:
```
seta = np.array([1, 2, 3, 4])
setb = np.array([3, 4, 5, 6])
union=np.union1d(seta,setb)
print('union set:',union)
```

```
union set: [1 2 3 4 5 6]
```

In [47]:
```
seta = np.array([1, 2, 3, 4])

setb = np.array([3, 4, 5, 6])
intersection=np.intersect1d(seta,setb)
print('intersection set:',intersection)
```

```
intersection set: [3 4]
```

ARRAY ATTRIBUTE FUNCTIONS

In [49]:
```
a= np.array([1, 2, 3, 4])
shape=a.shape
print('shape is :',shape)
```

```
shape is : (4,)
```

In [50]:
```
a= np.array([1, 2, 3, 4])
size=a.size
print('size is :',size)
```

```
size is : 4
```

In [51]:
```
a= np.array([1, 2, 3, 4])
dimensions =a.ndim
print('dimensions are:',dimensions)
```

```
dimensions are: 1
```

In [53]:
```
dtype=a.dtype
print('datatype is:',dtype)
```

```
datatype is: int64
```

COPY ARRAY

In [54]:
```
a= np.array([1, 2, 3, 4])
b=np.copy(a)
print('original array:',a)
print('copied array:',b)
```

```
original array: [1 2 3 4]
copied array: [1 2 3 4]
```

BYTES

In [55]:
```
a.nbytes
```

Out[55]: 32

In [56]:
```
b.nbytes
```

Out[56]: 32

sharing memory

In [58]:
```
shared_memory=np.shares_memory(a,b)
print('shared memory is:',shared_memory)
```

```
shared memory is: False
```

In [ ]: