1.Array Creation functions

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

In [12]:
```
# create an array from a list
a=np.array([1,2,3])
print("array a:",a)
```

```
array a: [1 2 3]
```

In [16]:
```
# create an array with evenly spaced values
b=np.arange(0,10,2)
print("array b:",b)
```

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

In [28]:
```
#create an array filled with zeroes
d=np.zeros((2,3))
print("array d:\n ",d)
```

```
array d:
  [[0. 0. 0.]
 [0. 0. 0.]]
```

In [30]:
```
# create an array filled with ones
e= np.ones((3,2))
print("array e:\n",e)
```

```
array e:
 [[1. 1.]
 [1. 1.]
 [1. 1.]]
```

In [34]:
```
# create an identity matrix
f = np.eye(4) # 4x4 identity matrix
print("Identity matrix f:\n", f)
```

```
Identity matrix f:
 [[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]
```

2. Array Manipulation Functions

In [37]:
```
# Reshape  an array
a1 = np.array([1,2,3])
reshaped =np.reshape(a1,(1,3))
print("Reshaped array:",reshaped)
```

```
Reshaped array: [[1 2 3]]
```

In [51]:
```
# Flatten an array
f1 = np.array([[1,2],[3,4]])
flattened = np.ravel(f1) #Flatten to 1D array
print("Flattened array :",flattened)
```

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

In [55]:
```
# Transpose an array
e1 =np.array([[1,2],[3,4]])
transposed = np.transpose(e1)# transpose the array
print("transposed arrays:\n",transposed)
```

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

In [57]:
```
# stack arrays vertically
a2=np.array([1,2])
b2=np.array([3,4])
stacked =np.vstack([a2,b2]) # stack a and b vertically
print("Stacked arrays :\n",stacked)
```

```
Stacked arrays :
 [[1 2]
 [3 4]]
```

3.Mathematical Functions

In [60]:
```
# add two  arrays
g = np.array([1,2,3,4])
added = np.add(g,2)
print("Added 2 to g :",added)
```

```
Added 2 to g : [3 4 5 6]
```

In [64]:
```
# square each element
squared =np.power(g,2) # square each element
print("Squared g:",squared)
```

```
Squared g: [ 1  4  9 16]
```

In [66]:
```
# square  root of each element
sqrt_val =np.sqrt(g) # square root of each element
print("square root of g :",sqrt_val)
```

```
square root of g : [1.         1.41421356 1.73205081 2.        ]
```

In [68]:
```
print(a1)
print(g)
```

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

In [72]:
```
# Dot product of two arrays
a2 =-np.array([1,2,3])
dot_product =np.dot(a2,g) # dot prodyuct of a and g
print("dot product of a and g:",dot_product)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[72], line 3
      1 # Dot product of two arrays
      2 a2 =-np.array([1,2,3])
----> 3 dot_product =np.dot(a2,g) # dot prodyuct of a and g
      4 print("dot product of a and g:",dot_product)

ValueError: shapes (3,) and (4,) not aligned: 3 (dim 0) != 4 (dim 0)
```

In [74]:
```
print(a)
print(a1)
```

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

In [76]:
```
a3=np.array([1,2,3])
dot_product = np.dot(a1,a) #dot product of a and g
print("Dot product of a1 and a:",dot_product)
```

```
Dot product of a1 and a: 14
```

4. Statistical Functions

In [79]:
```
s = np.array([1,2,3,4])
mean = np.mean(s)
print("mean of s:",mean)
```

```
mean of s: 2.5
```

In [81]:
```
# standard deviation of an array
std_dev = np.std(s)
print("Standard deviation of s :",std_dev)
```

```
Standard deviation of s : 1.118033988749895
```

In [85]:
```
# minimum element of an array
minimum = np.min(s)
print("Min of s:",minimum)
```

```
Min of s: 1
```

In [87]:
```
# maximum element of an array
maximum =np.max(s)
print("Max of s:",maximum)
```

```
Max of s: 4
```

5.Linear Algebra Functions

In [96]:
```
# create a matrix
matrix = np.array([[1,2],[3,4]])
```

In [98]:
```
# determinant of a matrix
determinant=np.linalg.det(matrix)
print("Determinant of matrix:",determinant)
```

```
Determinant of matrix: -2.0000000000000004
```

In [100…
```
# inverse of a matrix
inverse = np.linalg.inv(matrix)
print("Inverse of matrix:\n",inverse)
```

```
Inverse of matrix:
 [[-2.   1. ]
 [ 1.5 -0.5]]
```

6.Random Sampling Functions

In [103…
```
# generate random values between o and 1
random_vals = np.random.rand(3)
print("Random values :",random_vals)
```

```
Random values : [0.18725068 0.68911986 0.55047465]
```

In [105…
```
# set seed for reproducibility
np.random.seed(0)
```

In [107…
```
# Generate random values between o and 1
random_vals = np.random.rand(3) # array of 3 random values between 0 and 1
print("Random values :",random_vals)
```

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

In [111…
```
# Generate random integers
rand_ints =np.random.randint(0,10,size=5)
print("Random integers:",rand_ints)
```

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

In [113…
```
#  set seed for reproducibility
np.random.seed(0)
```

In [115…
```
# Generate random integers
rand_ints = np.random.randint(0,10,size=5) #random integers between 0 and 10
print("Random integers:",rand_ints)
```

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

7.Boolean & Logical Functions

In [119…
```
# check if all elements are True
# all
logical_test =np.array([True,False, True])
all_true = np.all(logical_test)
print("ALL elements True:",all_true)
```

```
ALL elements True: False
```

In [121…
```
# check if all elements are True
logical_test =np.array([True,False,True])
all_true =np.all(logical_test) # check if all are True
print("All elements True:",all_true)
```

```
All elements True: False
```

In [123…
```
# check if all elements are True
logical_test =np.array([False,False,False])
all_true =np.all(logical_test) # check if all are True
print("All elements True:",all_true)
```

```
All elements True: False
```

In [125…
```
# check if any elements are True
# any
any_true = np.any(logical_test) # check if any are True
print("Any elements True:",any_true)
```

```
Any elements True: False
```

8.Set Operations

In [131…
```
# Intersection of two arrays
set_a =np.array([1,2,3,4])
set_b =np.array([3,4,5,6])
intersection =np.intersect1d(set_a,set_b)
print("Intersection of a and b:",intersection)
```

```
Intersection of a and b: [3 4]
```

In [133…
```
# union of two arrays
union = np.union1d(set_a,set_b)
print("union of a and b:",union)
```

```
union of a and b: [1 2 3 4 5 6]
```

9.Array Attribute Functions

In [139…
```
# Array attributes
a=np.array([1,2,3])
shape =a.shape  # shape of the array
size =a.size   # number of elements
dimensions = a.ndim # number of dimenisons
dtype =a.dtype # data type of the array

print("shape of a:",shape)
print("size of a:",size)
print("Number of dimensions of a:",dimensions)
print("data type of a:",dtype)
```

```
shape of a: (3,)
size of a: 3
Number of dimensions of a: 1
data type of a: int32
```

10. Other Functions

In [142…
```
# create a copy of an array
a =np.array([1,2,3])
copied_array = np.copy(a) #create a copy of array
print("copied array:",copied_array)
```

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

In [144…
```
# size in bytes of an array
array_size_in_bytes =a.nbytes  # size in bytes
print("Size of a in bytes:",array_size_in_bytes)
```

```
Size of a in bytes: 12
```

In [146…
```
# check if two arrays share memory
shared = np.shares_memory(a,copied_array) #check if arrays share memory
print("Do a and copied_array share memory", shared)
```

```
Do a and copied_array share memory False
```

In [ ]:
```

```