Numpy Built-in Functions

1.Array Creations Functions

+ Code + Text

```
import numpy as np
```

```
#Creat an array from a list
a=np.array([1,2,3])
print('Array a:',a)
```

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

```
#Creating an array with evenly spaced values
b=np.arange(0,10,2) # Values from 0 to 10 with step 2
print ("Array b:",b)
```

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

```
#Create an array with linearly spaced values
c=np.linspace(0,1,5) # 5 values evenly spaced between 0 and 1
print("Array c:",c)
```

```
Array c: [0.   0.25 0.5  0.75 1.  ]
```

```
#Creating an array filled with zeros
d=np.zeros((2,3)) # 2x3 array of zeros
print("Array d:\n",d)
```

```
Array d:
 [[0. 0. 0.]
 [0. 0. 0.]]
```

```
# Create an array filled with ones
e=np.ones((4,3)) # 4x3 array of ones
print("array e:\n",e)
```

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

```
#Creating 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 Fuctions

```
# Reshaping an  array
a1=np.array([1,2,3])
reshaped=np.reshape(a1,(1,3))# Reshape to 1x3
print("Reshapped Array:",reshaped)
```

```
Reshapped Array: [[1 2 3]]
```

```
#Flattening 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]
```

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

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

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

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

3.Mathematical Functions

```
#Adding Two arrays
g=np.array([1,2,3,4])
added=np.add(g,2) #adding 2 to each element
print("Added 2 to g:",added)
```

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

```
#Square eache element
squared =np.power(g,2)#squaring each element
print("squared g:",squared)
```

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

```
#Square root of eache element
sqrt_val=np.sqrt(g)
print("sqrt value of g:",sqrt_val)
```

```
sqrt value of g: [1.         1.41421356 1.73205081 2.        ]
```

```
print(a1)
print(g)
```

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

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

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[16], line 3
      1 #Dot product of two arrays
      2 a2=np.array([1,2,3])
----> 3 dot_product=np.dot(a2,g)# Dot product 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)
```

```
print(a)
print(a1)
```

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

```
a3=np.array([1,2,3])
dot_prod=np.dot(a1,a)
print("Dot product od a1 and a:",dot_prod)
```

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

4. Statistical Functions

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

```
Mean of s: 2.5
```

```
# Standard deviation of an array
std_dev=np.std(s)
print("Standard deviation of s:",std_dev)
```

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

```
# Minimum element of an array
minimum=np.min(s)
print("Min of s:",minimum)
```

```
Min of s: 1
```

```
# Maximum elementof an array
maximum=np.max(s)
print("Max of s:",maximum)
```

```
Max of s: 4
```

5.Linear Algebra Fumctions

```
#Creating a matrix
matrix=np.array([[1,2],[3,4]])
```

```
# Determinant of matrix
determinant=np.linalg.det(matrix)
print("Determinant of matrix :",determinant)
```

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

```
# 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

```
#Generating random values between 0 and 1
random_vals=np.random.rand(3) #Array of 3 random values between 0 and 1
print("random values:",random_vals)
```

```
random values: [0.21911893 0.09059943 0.34731562]
```

```
#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: [1 1 7 5 1]
```

```
#Set seed for reproductibility
np.random.seed(0)
# 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

```
#Check if all elements are true
# all
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
```

```
#check if all elements are True
logical_test_1=np.array([False,False,False])
all_true=np.all(logical_test_1)#check if all True
print("All elements True:",all_true)
```

```
All elements True: False
```

```
#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

```
#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]
```

```
#Union of two arrays
union=np.union1d(set_a,set_b)
print("Union of a andb:",union)
```

```
Union of a andb: [1 2 3 4 5 6]
```

9. Array Attribute Functions

```
#Array attributes
a=np.array([1,2,3])
shape=a.shape #shape of a array
size=a.size #number of elements
dimensions=a.ndim #number of dimensions
stype=a.dtype#Data type of an array

print("Shape of a:",shape)
print("Size of a:",size)
print("Number of dimensions of a:",dimensions)
print("data type of a:",stype)
```

```
Shape of a: (3,)
Size of a: 3
Number of dimensions of a: 1
data type of a: int32
```

10. Other Functions

```
#Create a copy of an array
a=np.array([1,2,3])
copied_array=np.copy(a)
print("Copied array:",copied_array)
```

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

```
#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
```

```
#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?\n",shared)
```

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