Shubhangi_Dhikale_35

1. Create a null vector of size 10

In [3]:
```
import numpy as np
arr=np.zeros(10)
arr
```

Out[3]:
```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

2.How to find the memory size of an array

In [5]:
```
import numpy as np
arr=np.array([100,20,30])
print("size of the array:",arr.size)
print("Memory size of one array element in bytes:",arr.itemsize)
print("Memory size of numpy array in bytes:",arr.size*arr.itemsize)
```

```
size of the array: 3
Memory size of one array element in bytes: 4
Memory size of numpy array in bytes: 12
```

3.Create a null vector of size 10 but the fifth value which is 1

In [6]:
```
arr=np.zeros(10)
print(arr)
arr[5]=1
print("updated array:",arr)
```

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
updated array: [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
```

4.Create a vector with values ranging from 15 to 45

In [9]:
```
arr1=np.arange(15,45)
arr1
```

Out[9]:
```
array([15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31,
       32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44])
```

5.Reverse a vector (The first element becomes last)

In [13]:
```
arr1=np.arange(15,45)
print("original vector:",arr1)
reverse_vector=arr1[::-1]
print("reverse_vector:",reverse_vector)
```

```
original vector: [15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 3
5 36 37 38
 39 40 41 42 43 44]
reverse_vector: [44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24
23 22 21
 20 19 18 17 16 15]
```

6.Write a NumPy program to add, subtract, multiply, divide arguments element-wise

In [20]:
```
import numpy as np

print("Add")
print(np.add(5,6))
print("Subtract")
print(np.subtract(40,30))
print("multiply")
print(np.multiply(10,20))
print("divide")
print(np.divide(20,40))
```

```
Add
11
Subtract
10
multiply
200
divide
0.5
```

7.Write a NumPy program to round elements of the array to the nearest integer

In [23]:
```
import numpy as np
arr=np.array([-0.7,-1.5,0.5,0.8,-2.5,-1.9])
print("original array:")
print(arr)
arr=np.rint(arr)
print("Round element of the array to the nearest integer:")
print(arr)
```

```
original array:
[-0.7 -1.5  0.5  0.8 -2.5 -1.9]
Round element of the array to the nearest integer:
[-1. -2.  0.  1. -2. -2.]
```

8.Write a NumPy program to get the floor and ceiling values of the elements of a NumPy array

In [26]:
```
arr1=np.array([1.9,2.3,-1.7,9.0,-2.3])
print(np.ceil(arr1))
print(np.floor(arr1))
```

```
[ 2.  3. -1.  9. -2.]
[ 1.  2. -2.  9. -3.]
```

9.Write a NumPy program to calculate mean across dimensions, in a 2D NumPy array.

In [27]:
```
arr=np.array([[2,3,6,7,9]])
mean=np.mean(arr)
print(mean)
```

```
5.4
```

In [33]:
```
import numpy as np

arr=np.array([[5,6],[2,3]])
print(arr)
print("Mean of each column:",arr.mean(axis=0))
print("Mean of each row:",arr.mean(axis=1))
```

```
[[5 6]
 [2 3]]
Mean of each column: [3.5 4.5]
Mean of each row: [5.5 2.5]
```

10.Write a NumPy program to convert angles from degrees to radians for all elements in a given array.

In [36]:
```
arr1=np.array([0,30,45,60,90])
Radian=np.deg2rad(arr1)
Radian
```

Out[36]:
```
array([0.        , 0.52359878, 0.78539816, 1.04719755, 1.57079633])
```

11.What is the use of all and any function in numpy?

In [ ]:
```
uses of all and numpy:

1) all(): function returns True only if all items in a Numpy array evalute as Tru
2) any: function returns True if at least one item in a Numpy array evalutes to T
```

12.Create a 3x3 matrix with values ranging from 0 to 8

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

In [37]:
```
arr=np.arange(9).reshape(3,3)
arr
```

Out[37]:
```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

13.How to reverse the columns of a 2D array?

([[2, 1, 0], [5, 4, 3], [8, 7, 6]])

In [ ]:
```
```

14.How to reverse the rows of a 2D array?

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

In [ ]:
```
```

15.Find indices of non-zero elements from [1,2,0,0,4,0]

In [38]:
```
arr=np.nonzero([1,2,0,0,4,0])
arr
```

Out[38]:
```
(array([0, 1, 4], dtype=int64),)
```

16.Write a NumPy program to compute the determinant of an array.

In [40]:
```
import numpy as np
arr=np.array([[1,2],[3,4]])
print("original array:")
print(arr)
result=np.linalg.det(arr)
print("Determinant of the said array:")
print(result)
```

```
original array:
[[1 2]
 [3 4]]
Determinant of the said array:
-2.0000000000000004
```

17.Write a NumPy program to compute the inverse of a given matrix

In [41]:
```
import numpy as np
arr=np.array([[1,2],[3,4]])
print("original array:")
print(arr)
result=np.linalg.inv(arr)
print("Determinant of the said array:")
print(result)
```

```
original array:
[[1 2]
 [3 4]]
Determinant of the said array:
[[-2.   1. ]
 [ 1.5 -0.5]]
```

18.Create a random vector of size 30 and find the mean value

In [46]:
```
arr1=np.random.randint(1,50,size=30)
print(arr1)
print("mean value:",arr1.mean())
```

```
[24  3 41 22 10 11  1 11 12 49 44 23 22  7 31 25  7  7 27 42 41  5 22 27
 10 21 40 31 13 37]
mean value: 22.2
```

19.How to extract all numbers between a given range from a NumPy array?

In [48]:
```
import numpy as np
arr=np.arange(1,15)
index=np.where((arr>=5)&(arr<=10))
arr[index]
```

Out[48]:
```
array([ 5,  6,  7,  8,  9, 10])
```

20.Create a 3x3x3 array with random values

In [49]:
```
arr=np.random.randint(1,50,size=(3,3,3))
arr
```

Out[49]:
```
array([[[ 8,  1,  1],
        [15, 36, 26],
        [28, 12, 47]],

       [[12, 43, 15],
        [29, 25, 49],
        [49, 21, 12]],

       [[12, 13, 47],
        [34, 40, 35],
        [ 6,  6,  5]]])
```

In [52]:
```
arr=np.random.random((3,3,3))
arr
```

Out[52]:
```
array([[[0.45507016, 0.26383786, 0.54620499],
        [0.41692741, 0.75152119, 0.30881108],
        [0.91269296, 0.7773017 , 0.6560383 ]],

       [[0.0308917 , 0.25713842, 0.81006313],
        [0.23432063, 0.0722678 , 0.38486066],
        [0.04796813, 0.58874977, 0.63023331]],

       [[0.01227973, 0.5966161 , 0.92535678],
        [0.50981128, 0.20384141, 0.03839149],
        [0.78775965, 0.26494117, 0.87220617]]])
```

21.Create a 10x10 array with random values and find the minimum and maximum values

In [54]:
```
arr=np.random.randint(1,100,size=(10,10))
print(arr)
print()
print("maximum values from given array:",arr.max())
print("minimum values from given array:",arr.min())
```

```
[[ 1 56 19 90  1 21 55 29 67 37]
 [50 29 10 88 11 13  8 19 19 61]
 [23  9 70 48 96 19 82 67 41 81]
 [ 2 30 44 86 78 29 68 49 81 35]
 [28 74 79 98 90 59  9 86 42 77]
 [79 39 95 21 77 19 76 78 68 49]
 [26 91 73 56 67 29 71 58 68 21]
 [79 73 39 43 49  9 97 19 72 69]
 [35 76 43 44 42 50 44 46 38 16]
 [11 67 69 16 13 90 13 87 76 59]]

maximum values from given array: 98
minimum values from given array: 1
```

In [55]:
```
arr=np.random.random((10,10))
print(arr)
print()
print("maximum values from given array:",arr.max())
print("minimum values from given array:",arr.min())
```

```
[[0.63641904 0.11523437 0.70267222 0.66385866 0.01924557 0.43082195
  0.18756429 0.14546045 0.76459772 0.46657649]
 [0.47109929 0.01562405 0.28972219 0.06210909 0.91701887 0.75974713
  0.09934166 0.39198767 0.79170962 0.17378588]
 [0.14962925 0.02415602 0.6711045  0.50340764 0.22563459 0.66457423
  0.80557391 0.96505762 0.08020143 0.29389057]
 [0.56728018 0.12055227 0.84823172 0.94553115 0.26135247 0.00663115
  0.03393952 0.43467732 0.14469541 0.22465989]
 [0.6155919  0.30642052 0.53659143 0.45658886 0.00311525 0.98663214
  0.93085298 0.11620872 0.64671042 0.54003447]
 [0.65020799 0.68904673 0.17373358 0.65159593 0.7830852  0.47737532
  0.60336155 0.76269524 0.87305977 0.63421899]
 [0.96764745 0.59203072 0.90920874 0.02664744 0.83839752 0.89064572
  0.65607391 0.4695312  0.06298954 0.9472745 ]
 [0.8880891  0.42706545 0.70943368 0.3761936  0.55495817 0.77683031
  0.39913402 0.15642457 0.0501205  0.4325203 ]
 [0.8389243  0.88358373 0.42638031 0.47444406 0.61564871 0.51476241
  0.22006702 0.42733861 0.33551346 0.45721753]
 [0.61037369 0.78465441 0.63287193 0.76665197 0.58970128 0.80839644
  0.68066115 0.98461959 0.41330348 0.57596578]]

maximum values from given array: 0.9866321390158522
minimum values from given array: 0.0031152522650774728
```

22.Create a 2d array with 1 on the border and 0 inside

In [56]:
```
import numpy as np
arr=np.ones((4,4))
print("original array:")
print(arr)
print("1 on the border and 0 inside in the array")
arr[1:-1,1:-1]=0
print(arr)
```

```
original array:
[[1. 1. 1. 1.]
 [1. 1. 1. 1.]
 [1. 1. 1. 1.]
 [1. 1. 1. 1.]]
1 on the border and 0 inside in the array
[[1. 1. 1. 1.]
 [1. 0. 0. 1.]
 [1. 0. 0. 1.]
 [1. 1. 1. 1.]]
```

23.Create a 5x5 matrix with values 1,2,3,4 just below the diagonal

In [57]:
```
arr=np.diag(1+np.arange(4),k=-1)
print(arr)
```

```
[[0 0 0 0 0]
 [1 0 0 0 0]
 [0 2 0 0 0]
 [0 0 3 0 0]
 [0 0 0 4 0]]
```

24.Create a 3x3 identity matrix

In [59]:
```
arr=np.identity(3,)
arr
```

Out[59]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

In [60]:
```
arr=np.eye(3,)
arr
```

Out[60]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

25.Create a 8x8 matrix and fill it with a checkerboard pattern

In [61]:
```
arr=np.zeros((8,8),dtype=int)
arr[1::2,::2]=1
arr[::2,1::2]=1
print(arr)
```

```
[[0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]]
```

26.Multiply a 5x3 matrix by a 3x2 matrix (real matrix product)

In [62]:
```
arr=np.dot(np.ones((5,3)),np.ones((3,2)))
print(arr)
```

```
[[3. 3.]
 [3. 3.]
 [3. 3.]
 [3. 3.]
 [3. 3.]]
```

In [64]:
```
arr1=np.random.randint(1,20,size=(5,3))
print(arr1)
arr2=np.random.randint(1,30,size=(3,2))
print(arr2)
new_arr=np.dot(arr1,arr2)
print(new_arr)
```

```
[[19 17  1]
 [13 16  1]
 [15 17 14]
 [ 6  6 19]
 [16  4  2]]
[[15 19]
 [ 2 25]
 [16 13]]
[[335 799]
 [243 660]
 [483 892]
 [406 511]
 [280 430]]
```

27.Given a 1D array, negate all elements which are between 3 and 8, in place

In [ ]:
```

```

28.How to round away from zero a float array?

In [67]:
```
arr=np.array([1.,2.,3.,4.,5.])
arr1=np.ceil(arr)
arr1
```

Out[67]:
```
array([1., 2., 3., 4., 5.])
```

29.How to find common values between the two arrays

In [68]:
```
arr1=np.array([20,10,30,40,50])
arr2=np.array([10,60,50,30,70])
arr=[]
for i in arr1:
    for j in arr2:
        if i==j:
            arr.append(i)
print(arr)
```

```
[10, 30, 50]
```

In [70]:
```
arr1=np.array([20,10,30,40,50])
arr2=np.array([10,60,50,30,70])
new_arr=np.intersect1d(arr1,arr2)
print(new_arr)
```

```
[10 30 50]
```

30.Create a vector of size 10 with values ranging from 0 to 1, both excluded

In [77]:
```
arr1=np.linspace(0,1,12)
print(arr1)
print()
print(arr1[1:-1])
```

```
[0.         0.09090909 0.18181818 0.27272727 0.36363636 0.45454545
 0.54545455 0.63636364 0.72727273 0.81818182 0.90909091 1.        ]

[0.09090909 0.18181818 0.27272727 0.36363636 0.45454545 0.54545455
 0.63636364 0.72727273 0.81818182 0.90909091]
```

31.Create a random vector of size 10 and sort it

In [83]:
```
arr1=np.random.randint(1,50,size=10)
print(arr1)
print("sorted array:")
arr1.sort()
print(arr1)
```

```
[ 8 29  7 11 40 28 20 27 49 18]
sorted array:
[ 7  8 11 18 20 27 28 29 40 49]
```

32.Create a 5x5 matrix with row values ranging from 0 to 4

In [86]:
```
arr=np.zeros((5,5))
print("original array:")
print(arr)
print("Row values ranging from 0 to 4:")
arr+=np.arange(5)
print(arr)
```

```
original 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.]]
Row values ranging from 0 to 4:
[[0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]]
```

33.Consider two random arrays A and B, check if they are equal.

In [87]:
```
import numpy as np

arr1=np.random.randint(0,2,6)
print("First array:")
print(arr1)
arr2=np.random.randint(0,2,6)
print("Second array:")
print(arr2)
print("Test above two arrays are equal or not")
arr_equal=np.allclose(arr1,arr2)
print(arr_equal)
```

```
First array:
[0 0 1 0 1 0]
Second array:
[1 1 0 1 0 0]
Test above two arrays are equal or not
False
```

In [91]:
```
arrA=np.random.randint(0,2,6)
print("First array:")
print(arr1)
arrB=np.random.randint(0,2,6)
print("Second array:")
print(arr2)
if arrA.all()==arrB.all():
    print("Two array are equal")
else:
    print("Two array are not equal")
```

```
First array:
[0 0 1 0 1 0]
Second array:
[1 1 0 1 0 0]
Two array are equal
```

34.Create a random vector of size 10 and replace the maximum value by 0

In [99]:
```
arr=np.random.randint(1,20,size=10)
print(arr)
print("after sorting array:")
arr.sort()
print(arr)
print("Maximum value replace by 0:")
arr[9]=0
print(arr)
```

```
[ 9  3  1 17 10  3 11  3 11 17]
after sorting array:
[ 1  3  3  3  9 10 11 11 17 17]
Maximum value replace by 0:
[ 1  3  3  3  9 10 11 11 17  0]
```

35.How to find out multiple indices of an item?

In [ ]:
```

```

36.What is the equivalent of enumerate for NumPy arrays?

In [ ]:
```

```

37.How to sort an array by the nth column?

In [100]:
```
arr=np.random.randint(0,10,(3,3))
print(arr)
print("sort an array by the nth column:")
print(arr[arr[:,1].argsort()])
```

```
[[8 8 8]
 [0 9 4]
 [4 4 9]]
sort an array by the nth column:
[[4 4 9]
 [8 8 8]
 [0 9 4]]
```

38.How to swap two rows of an array?

In [104]:
```
arr=np.array([[2,3,4],[6,7,8],[4,5,6]])
print(arr)
print("swap two rows:")
arr[[0,1]]=arr[[1,0]]
print(arr)
```

```
[[2 3 4]
 [6 7 8]
 [4 5 6]]
swap two rows:
[[6 7 8]
 [2 3 4]
 [4 5 6]]
```

39.How to compute the mean of a NumPy array?

In [105]:
```
arr1=np.array([10,20,30,40,50])
mean=np.mean(arr1)
print(mean)
```

```
30.0
```

40.How to compute the median of a NumPy array?

In [108]:
```
arr1=np.array([3,1,5,6,2,9,8])
median=np.median(arr1)
print("median of Numpy array:")
print(median)
```

```
median of Numpy array:
5.0
```

41.How to compute the standard deviation of a NumPy array?

In [109]:
```
arr1=np.array([4,5,2,6,7,8])
std=np.std(arr1)
print("standard deviation of given array:")
print(std)
```

```
standard deviation of given array:
1.9720265943665387
```

42.How to compute the mode of a NumPy array?

In [113]:
```
from scipy import stats
arr=np.array([0,0,1,1,1,1,2,2,3])
print(arr)
mode=stats.mode(arr)
print("mode of given array:")
print(mode)
```

```
[0 0 1 1 1 1 2 2 3]
mode of given array:
ModeResult(mode=array([1]), count=array([4]))
```

In [112]:
```
from scipy import stats
arr=np.array([[2,3,7],[5,6,7],[1,1,5]])
print(arr)
mode=stats.mode(arr)
print("mode of given array:")
print(mode)
```

```
[[2 3 7]
 [5 6 7]
 [1 1 5]]
mode of given array:
ModeResult(mode=array([[1, 1, 7]]), count=array([[1, 1, 2]]))
```

43.How to print only 3 decimal places in a python NumPy array?

In [121]:
```
rand_arr=np.random.randn(5)
print(rand_arr)
print("3 decimal numberin given array:")
new_arr=np.round(rand_arr,3)
print(new_arr)
```

```
[ 0.60431603 -1.61125455  0.76646582  0.62010512  0.64153391]
3 decimal numberin given array:
[ 0.604 -1.611  0.766  0.62   0.642]
```

44.Write a NumPy program to compute the inverse of a given matrix

In [127]:
```
arr=np.array([[2,3,4],[6,7,8],[1,2,9]])
print("original array:\n",arr)
new_arr=np.linalg.inv(arr)
print("inverse of a given matrix:\n",new_arr)
```

```
original array:
 [[2 3 4]
 [6 7 8]
 [1 2 9]]
inverse of a given matrix:
 [[-1.95833333  0.79166667  0.16666667]
 [ 1.91666667 -0.58333333 -0.33333333]
 [-0.20833333  0.04166667  0.16666667]]
```

45.Write a NumPy program to compute the covariance matrix of two given arrays

In [130]:
```
import numpy as np
arr1=np.array([0,1,2])
arr2=np.array([2,1,0])
print("\noriginal array1:",arr1)
print("\noriginal array2:",arr2)
cov=np.cov(arr1,arr2)
print("\ncovariance matrix of two given arrays:\n",cov)
```

```
original array1: [0 1 2]

original array2: [2 1 0]

covariance matrix of two given arrays:
 [[ 1. -1.]
 [-1.  1.]]
```

46.How to find the most frequent value in a NumPy array?

In [131]:
```
from scipy import stats
arr=np.array([0,1,1,2,2,2])
mode=stats.mode(arr)
print("most frequentvalue:",mode)
```

```
most frequentvalue: ModeResult(mode=array([2]), count=array([3]))
```

47.How to convert 1D array to 3D array?

In [132]:
```
arr=np.array([1,2,3,4,5],ndmin=3)
print(arr)
```

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

48.How to convert 4D array to 2D array?

In [134]:
```
arr1=np.array([[[[4,5,6],[7,8,9],[3,4,5]]]],ndmin=2)
print(arr1)
```

```
[[[[4 5 6]
   [7 8 9]
   [3 4 5]]]]
```

49.Create a Numpy array filled with all zeros

In [137]:
```
arr=np.zeros((4,5),dtype=int)
print(arr)
```

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

50.Find the number of rows and columns of a given matrix using NumPy

In [139]:
```
arr=np.array([[5,6],[7,8]])
print(arr)
print("number of rows and columns of given matrix:",arr.shape)
```

```
[[5 6]
 [7 8]]
number of rows and columns of given matrix: (2, 2)
```