1. Create a null vector of size 10

In [154]:
```python
import numpy as np

vector = np.zeros(10)
print(vector)

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

2. How to find the memory size of an array

In [5]:
```python
import numpy as np

array = np.array([100,200,300,400,500])
print("Memory size of an array is:",array.size)

```
```
Memory size of an array is: 5
```

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

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

null_vector = np.zeros(10)
null_vector[5]=1
print(null_vector)



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

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

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

vector = np.arange(15,45)
print(vector)

```
```
[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 [28]:
```python
import numpy as np

vector = np.arange(1,11)
for i in vector[::-1]:
    print(i)



```

```
10
9
8
7
6
5
4
3
2
1
```

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

In [31]:
```python
X = np.array([5,10,15,20])
Y = np.array([2,4,6,8])

Addition = X+Y
Subtraction = X-Y
Multiplication = X*Y
Division = X/Y

print("Addition of elements:",Addition)
print("Subtraction of elements:",Subtraction)
print("Multiply of elements:",Multiplication)
print("Division of elements:",Division)
```

```
Addition of elements: [ 7 14 21 28]
Subtraction of elements: [ 3  6  9 12]
Multiply of elements: [ 10  40  90 160]
Division of elements: [2.5 2.5 2.5 2.5]
```

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

In [3]:
```python
import numpy as np

arr = np.array([2.1, 2.5, 4.5, 2.9, 5.1, -3.5, -5.9])
r1 = np.around(arr)
r1
```

Out[3]:
```
array([ 2.,  2.,  4.,  3.,  5., -4., -6.])
```

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

In [54]:
```
import numpy as np
array1=np.ceil(9.3)  #grater nearest

array2=np.floor(7.3)  #lowest nearest

print("ceil value of element:",array1)
print("floor value of element:",array2)
```

```
ceil value of element: 10.0
floor value of element: 7.0
```

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

In [61]:
```
array = np.array([[10,20,30,40],[4,5,6,7]])
print(array)

mean=np.mean(array)
print(" mean values is:",mean)
```

```
[[10 20 30 40]
 [ 4  5  6  7]]
 mean values is: 15.25
```

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

In [4]:
```
array= np.array([7,5,9,3,6])
print(array)

radian=np.deg2rad(array)
radian
```

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

Out[4]:
```
array([0.12217305, 0.08726646, 0.15707963, 0.05235988, 0.10471976])
```

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

[[0 1 2]

[3 4 5]

[6 7 8]]

In [71]:
```
array=np.arange(0,9)
print("array is:",array)

array1=array.reshape(3,3)
print(array1)
```

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

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

[[6, 7, 8],

[3, 4, 5],

[0, 1, 2]]

In [9]:
```
array=np.array([[6,7,8],[3,4,5],[0,1,2]])
array
reverse=array[::-1]
reverse
```

Out[9]:
```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

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

In [93]:
```
import numpy as np
a =np.array([[1,2],[3,4]])
b =np.array([5,6])

result =np.linalg.solve(a,b)
result
```

Out[93]:
```
array([-4. ,  4.5])
```

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

In [105]:
```
A=np.array([[3,5],[4,2]])
print(A)
result=np.linalg.inv(A)
print(result)
```

```
[[3 5]
 [4 2]]
[[-0.14285714  0.35714286]
 [ 0.28571429 -0.21428571]]
```

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

In [109]:
```python
array1=np.random.rand(30)
print(array1)

mean=np.mean(array1)
print("mean value is:",mean)
```

```
[0.78335199 0.54467336 0.77768188 0.29502972 0.65530658 0.21251018
 0.63237935 0.00965015 0.60870707 0.73780947 0.11420345 0.87498145
 0.64330383 0.53406554 0.31255398 0.47279753 0.33558969 0.35814417
 0.36231065 0.74165381 0.44143996 0.57755708 0.46614109 0.81466294
 0.16816142 0.68051172 0.79538873 0.26641931 0.63203536 0.5171634 ]
mean value is: 0.5122061620931817
```

16. Create a 3x3x3 array with random values

In [111]:
```python
import numpy as np
array = np.random.rand(3,3,3)
print(array)

```

```
[[[0.90980402 0.97409356 0.89048663]
  [0.4105884  0.71703997 0.36195031]
  [0.51883398 0.27374952 0.52900041]]

 [[0.56535376 0.54093325 0.3169943 ]
  [0.82721609 0.95305789 0.92559322]
  [0.63650941 0.53972216 0.80821613]]

 [[0.40636713 0.49462159 0.27713879]
  [0.83230679 0.4724958  0.32783568]
  [0.42065858 0.62009755 0.83859254]]]
```

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

In [116]:
```
array=np.random.rand(10,10)
print(array)

Min=np.min(array)
Max=np.max(array)
print("minimum value is:",Min)
print("maximum value is:",Max)
```

```
[[0.67803617 0.93414145 0.76371185 0.07866413 0.40547236 0.88386773
  0.06739827 0.78487061 0.31816834 0.10743118]
 [0.32687564 0.54538622 0.94546183 0.58971904 0.65263144 0.74919179
  0.78209556 0.49903902 0.19599686 0.7164744 ]
 [0.72766518 0.90077134 0.14971169 0.7397309  0.11288749 0.01214059
  0.65731138 0.66998226 0.39900868 0.16080371]
 [0.49536216 0.33030075 0.92252331 0.37993164 0.07507032 0.16406658
  0.64132255 0.45928109 0.02023514 0.30240114]
 [0.14232211 0.444684   0.2444666  0.7792428  0.64890621 0.6807941
  0.46940619 0.24769539 0.87661653 0.35209528]
 [0.56553096 0.1015039  0.56670005 0.29067505 0.40870211 0.43671059
  0.06904399 0.82113808 0.04735976 0.9592013 ]
 [0.34567406 0.42369447 0.48422294 0.59400652 0.68154547 0.18413659
  0.47287317 0.82878016 0.16154175 0.18417093]
 [0.13861389 0.30748401 0.19646204 0.70228264 0.23012746 0.05462917
  0.74833632 0.33036364 0.79615452 0.99760908]
 [0.43713637 0.39393591 0.66868743 0.00931782 0.81780172 0.39938697
  0.21555574 0.16233507 0.91619391 0.56881617]
 [0.39619162 0.12309978 0.97215246 0.19458346 0.72472163 0.07815142
  0.5708394  0.76457686 0.6309868  0.0697517 ]]
minimum value is: 0.00931781578771207
maximum value is: 0.9976090842061924
```

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

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

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

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

In [126]:
```
rows = 10
column = 10
array = np.zeros((rows, column))
array[0,:] = 1
array[-1,:] = 1
array[:,0] = 1
array[:,-1] = 1

print(array)

```

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

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

In [139]:
```
array=np.diag([1,2,3,4,5])
print(array)
```

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

20. Create a 3x3 identity matrix

In [133]:
```
arr1=np.eye(3,3)
print(arr1)

```

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

21. How to find common values between the two arrays?

In [140]:
```python
array1=np.array([1, 2, 3, 4, 5])
array2=np.array([3, 6, 7, 5, 4])

array3=np.intersect1d(array1,array2)

print("Common values is:",array3)
```

```
Common values is: [3 4 5]
```

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

In [152]:
```python
array=np.random.randint(1,30,size=10)
print(array)

array1=np.sort(array)
print("array1 is:",array1)
```

```
[19 14  9 13  2  1 21 15 10 24]
array1 is: [ 1  2  9 10 13 14 15 19 21 24]
```

23. Create a 5x5 matrix with 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.]]

In [162]:
```python
array=np.zeros((5,5))
array += np.arange(5)
print(array)
```

```
[[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.]]
```

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

In [11]:
```
Arr1=np.array([2,3,4])
Arr2=np.array([4,2,3])
if np.array_equal(Arr1,Arr2):
    print("Array are equal")
else:
    print("Array are not equal")


```

```
Array are not equal
```

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

In [10]:
```
array= np.random.randint(20,40,size=10)

print(array)
array[array.argmax()] = 0

print("Replace the maximum value by 0:")
print(array)
```

```
[37 25 30 37 21 27 25 35 27 22]
Replace the maximum value by 0:
[ 0 25 30 37 21 27 25 35 27 22]
```

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

In [ ]:
```
The NumPy equivalent of enumerate is ndenumerate.
```

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

In [17]:
```
import numpy as np
n = np.random.randint(0, 10, (3, 3))
sorted_n= n[n[:, 1].argsort()]
print(sorted_n)
```

```
[[8 1 8]
 [9 2 1]
 [5 8 7]]
```

28. How to swap two rows of an array?

In [26]:
```
import numpy as np
array = np.random.randint(0,10,(3,3))
print("original array")
print(array)

array[[0,2]] == array[[2,0]]
array
```

```
original array
[[0 3 3]
 [6 0 4]
 [4 3 6]]
```

Out[26]:
```
array([[0, 3, 3],
       [6, 0, 4],
       [4, 3, 6]])
```

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

In [ ]:
```
using mean fuction.
```

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

In [27]:
```
import numpy as np
array= np.array([1,2,3,4,5,6,7,8,9,10])
median_value = np.median(array)
median_value
```

Out[27]:
```
5.5
```

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

In [28]:
```
import numpy as np
array=([1,2,3,4,5,6,7,8,9,10])
value=np.std(array)
value
```

Out[28]:
```
2.8722813232690143
```

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

In [32]:
```
import numpy as np
from scipy.stats import mode
array=([1,2,3,4,5,6,7,8,9,10])
value=mode(array)
value
```

Out[32]:
```
ModeResult(mode=1, count=1)
```

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

In [33]:
```
import numpy as np
n = np.random.random((3,3))
print(n)
n = np.round_(n,decimals = 3)
n

```

```
[[0.56151106 0.01016846 0.89802477]
 [0.52048932 0.75652653 0.33265059]
 [0.10698779 0.9944766  0.2487024 ]]
```

Out[33]:
```
array([[0.562, 0.01 , 0.898],
       [0.52 , 0.757, 0.333],
       [0.107, 0.994, 0.249]])
```

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

In [34]:
```
import numpy as np
n = np.random.randint(1,9,(3,3))
print("origial array: ",n)
n = np.linalg.inv(n)
print("inverse: ",n)
```

```
origial array:  [[3 5 3]
 [4 2 2]
 [4 5 8]]
inverse:  [[-0.09090909  0.37878788 -0.06060606]
 [ 0.36363636 -0.18181818 -0.09090909]
 [-0.18181818 -0.07575758  0.21212121]]
```

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

In [36]:
```
import numpy as np
array1 = np.array([1, 2, 3, 4, 5])
array2 = np.array([5, 4, 3, 2, 1])
covariance_matrix = np.cov(array1, array2)
covariance_matrix
```

Out[36]:
```
array([[ 2.5, -2.5],
       [-2.5,  2.5]])
```

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

```
Find the most frequent value (mode) in a NumPy array, you can use
NumPy and Python's standard library functions.
```

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

In [58]:
```
import numpy as np
n = np.array([1,2,3,4,5,6,7,8,9,10,11,12])
n = n.reshape(2,2,3)
n
```

Out[58]:
```
array([[[ 1,  2,  3],
        [ 4,  5,  6]],

       [[ 7,  8,  9],
        [10, 11, 12]]])
```

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

In [69]:
```
import numpy as np
n = np.array([1,2,3,4,5,6,7,8,9,10,11,12])
n = n.reshape(2,2,3)
print("original array :" ,n)
k= n.reshape(3,4)
print("4D array:" ,k)
```

```
original array : [[[ 1  2  3]
  [ 4  5  6]]

 [[ 7  8  9]
  [10 11 12]]]
4D array: [[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]
```

39. Create a Numpy array filled with all zeros

In [71]:
```
array=np.zeros(10)
array
```

Out[71]:
```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

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

In [73]:
```
array= np.random.randint(1,10,(3,3))
array=np.shape(array)
array
```

Out[73]:
```
(3, 3)
```

In [ ]:
```

```