**1. Import numpy as np and see the version**

Q. Import numpy as np and print the version number.

import numpy as np

print(np.\_\_version\_\_)

#> 1.13.3

**2. How to create a 1D array?**

Q. Create a 1D array of numbers from 0 to 9

Desired output:

#> array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

arr = np.arange(10)

arr

#> array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

## 3. How to create a boolean array?

Q. Create a 3×3 numpy array of all True’s

np.full((3, 3), True, dtype=bool)

#> array([[ True, True, True],

#> [ True, True, True],

#> [ True, True, True]], dtype=bool)

# Alternate method:

np.ones((3,3), dtype=bool)

## 4. How to extract items that satisfy a given condition from 1D array?

Q. Extract all odd numbers from arr

Input:

arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

Desired output:

#> array([1, 3, 5, 7, 9])

# Input

arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

# Solution

arr[arr % 2 == 1]

#> array([1, 3, 5, 7, 9])

## 5. How to replace items that satisfy a condition with another value in numpy array?

Q. Replace all odd numbers in arr with -1

Input:

arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

Desired Output:

#> array([ 0, -1, 2, -1, 4, -1, 6, -1, 8, -1])

arr[arr % 2 == 1] = -1

arr

#> array([ 0, -1, 2, -1, 4, -1, 6, -1, 8, -1])

**6. How to replace items that satisfy a condition without affecting the original array?**

Q. Replace all odd numbers in arr with -1 without changing arr

Input:

arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

Desired Output:

out

#> array([ 0, -1, 2, -1, 4, -1, 6, -1, 8, -1])

arr

#> array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

Solution

import numpy as np

arr = np.arange(10)

out = np.where(arr % 2 == 1, -1, arr)

print(out)

print(arr)

**7. How to reshape an array?**

Q. Convert a 1D array to a 2D array with 2 rows

Input:

np.arange(10)

#> array([ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

Desired Output:

#> array([[0, 1, 2, 3, 4],

#> [5, 6, 7, 8, 9]])

arr = np.arange(10)

arr.reshape(2, -1) # Setting to -1 automatically decides the number of cols

#> array([[0, 1, 2, 3, 4],

#> [5, 6, 7, 8, 9]])

**How to extract all numbers between a given range from a numpy array?**

Q. Get all items between 5 and 10 from a.

Input:

a = np.array([2, 6, 1, 9, 10, 3, 27])

Desired Output:

(array([6, 9, 10]),)

a = np.arange(15)

# Method 1

index = np.where((a >= 5) & (a <= 10))

a[index]

# Method 2:

index = np.where(np.logical\_and(a>=5, a<=10))

a[index]

#> (array([6, 9, 10]),)

# Method 3: (thanks loganzk!)

a[(a >= 5) & (a <= 10)]

**16. How to swap two columns in a 2d numpy array?**

Q. Swap columns 1 and 2 in the array arr.

arr = np.arange(9).reshape(3,3)

arr

Show Solution

# Input

arr = np.arange(9).reshape(3,3)

arr

# Solution

arr[:, [1,0,2]]

#> array([[1, 0, 2],

#> [4, 3, 5],

#> [7, 6, 8]])

**17. How to swap two rows in a 2d numpy array?**

Q. Swap rows 1 and 2 in the array arr:

arr = np.arange(9).reshape(3,3)

arr

# Input

arr = np.arange(9).reshape(3,3)

# Solution

arr[[1,0,2], :]

#> array([[3, 4, 5],

#> [0, 1, 2],

#> [6, 7, 8]])

**How to print only 3 decimal places in python numpy array?**

Q. Print or show only 3 decimal places of the numpy array rand\_arr.

rand\_arr = np.random.random((5,3))

# Input

rand\_arr = np.random.random((5,3))

# Create the random array

rand\_arr = np.random.random([5,3])

# Limit to 3 decimal places

np.set\_printoptions(precision=3)

rand\_arr[:4]

#> array([[ 0.443, 0.109, 0.97 ],

#> [ 0.388, 0.447, 0.191],

#> [ 0.891, 0.474, 0.212],

#> [ 0.609, 0.518, 0.403]])

**How to get the common items between two python numpy arrays?**

Q. Get the common items between a and b

Input:

a = np.array([1,2,3,2,3,4,3,4,5,6])

b = np.array([7,2,10,2,7,4,9,4,9,8])

Desired Output:

array([2, 4])

a = np.array([1,2,3,2,3,4,3,4,5,6])

b = np.array([7,2,10,2,7,4,9,4,9,8])

c=np.intersect1d(a,b)

#> array([2, 4])

**How to remove from one array those items that exist in another?**

Q. From array a remove all items present in array b

Input:

a = np.array([1,2,3,4,5])

b = np.array([5,6,7,8,9])

Desired Output:

array([1,2,3,4])

a = np.array([1,2,3,4,5])

b = np.array([5,6,7,8,9])

# From 'a' remove all of 'b'

c=np.setdiff1d(a,b)

#> array([1, 2, 3, 4])

**How to get the positions where elements of two arrays match?**

Q. Get the positions where elements of a and b match

Input:

a = np.array([1,2,3,2,3,4,3,4,5,6])

b = np.array([7,2,10,2,7,4,9,4,9,8])

Desired Output:

#> (array([1, 3, 5, 7]),)

a = np.array([1,2,3,2,3,4,3,4,5,6])

b = np.array([7,2,10,2,7,4,9,4,9,8])

np.where(a == b)

#> (array([1, 3, 5, 7]),)

**How to extract all numbers between a given range from a numpy array?**

Q. Get all items between 5 and 10 from a.

Input:

a = np.array([2, 6, 1, 9, 10, 3, 27])

Desired Output:

(array([6, 9, 10]),)

a = np.arange(15)

# Method 1

index = np.where((a >= 5) & (a <= 10))

a[index]

# Method 2:

index = np.where(np.logical\_and(a>=5, a<=10))

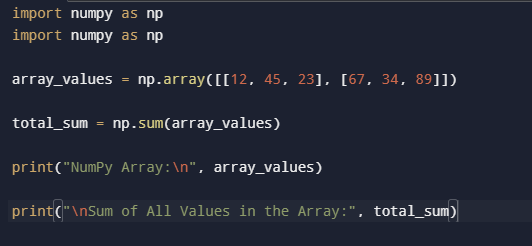
a[index]

#> (array([6, 9, 10]),)

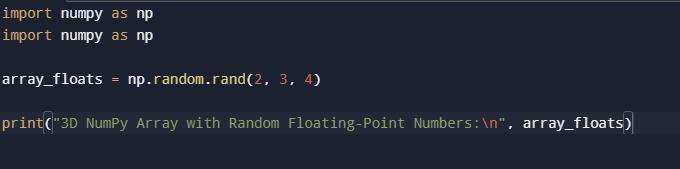
# Method 3: (thanks loganzk!)

a[(a >= 5) & (a <= 10)]

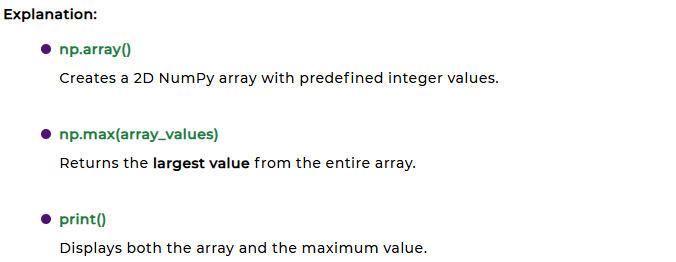
**Q.Find the Sum of All Values in a NumPy Array**

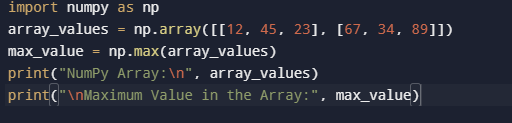


**Q.Create a NumPy Array of Floating-Point Numbers Between 0 and 1 with Shape (2, 3, 4).**

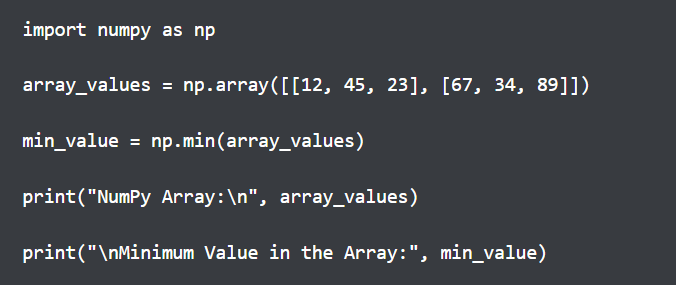


Q. **Find the Maximum Value in a NumPy Array**

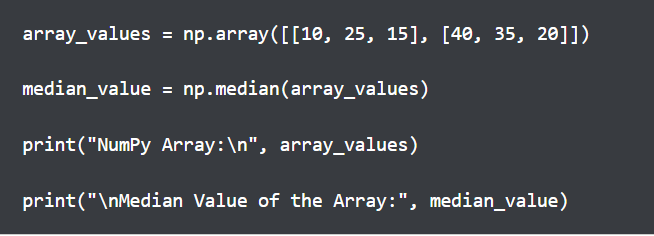
****



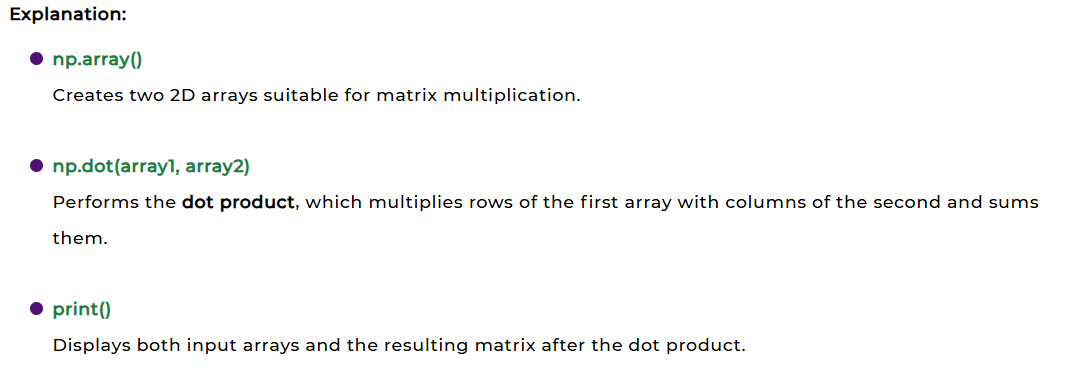
Q. **Find the Minimum Value in a NumPy Array**

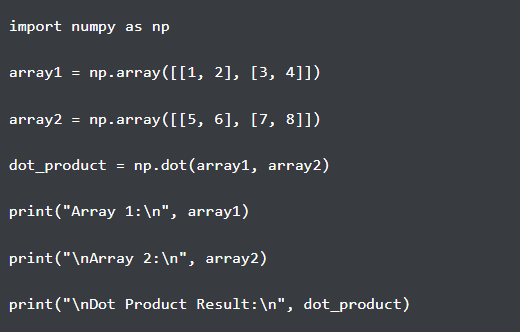


Q. **Find the Median Value of a NumPy Array**

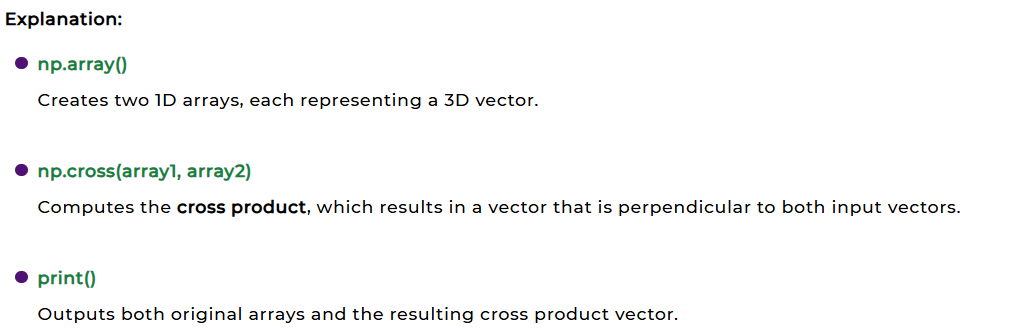


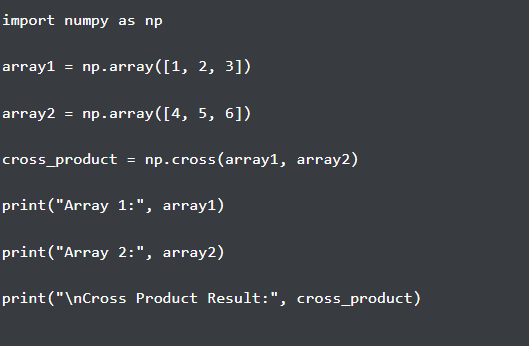
Q. **Compute the Dot Product of Two NumPy Arrays**



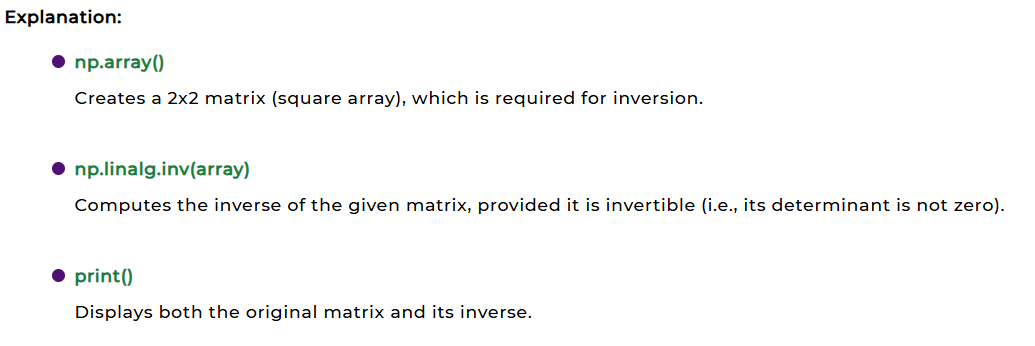


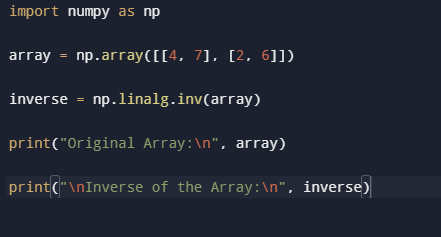
Q. **Compute the Cross Product of Two NumPy Arrays**

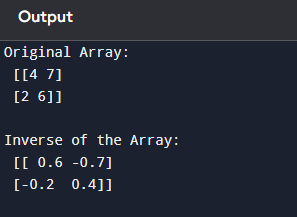




Q. **Compute the Inverse of a NumPy Array**







### Q. Reversed Array

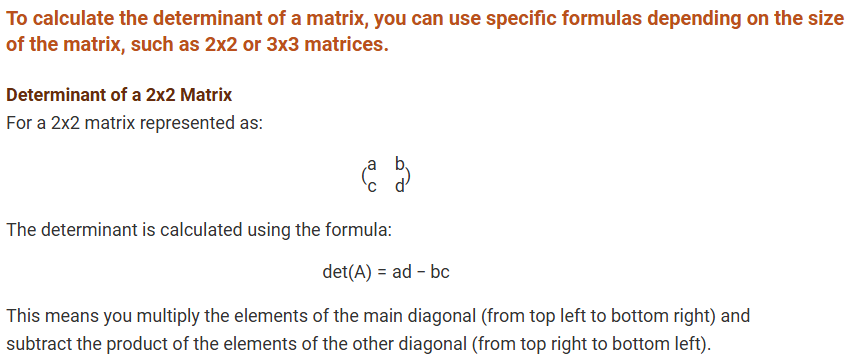
### Example:

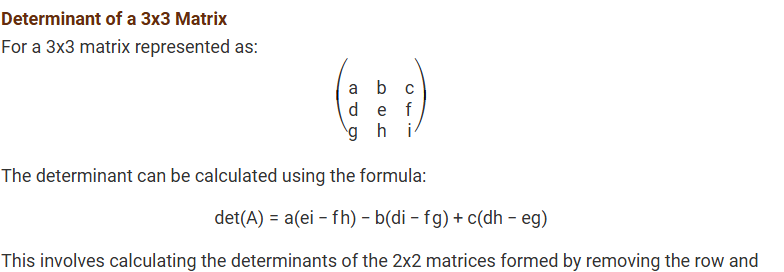
### 

### Output:

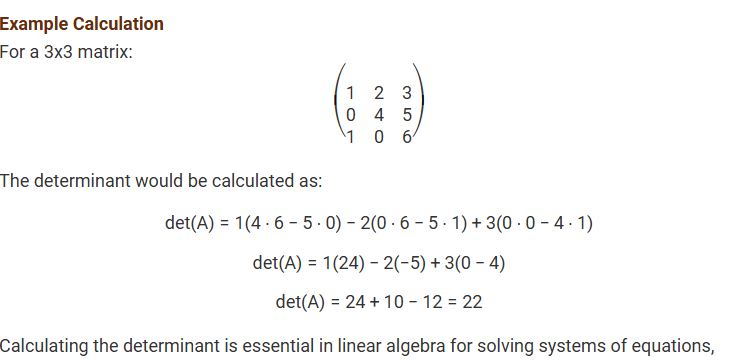
### [4 3 2 1]

Q. **Compute the Determinant of a NumPy Array**

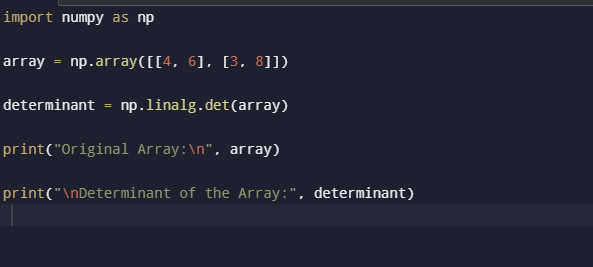


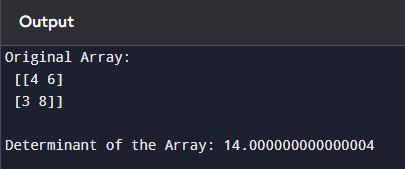






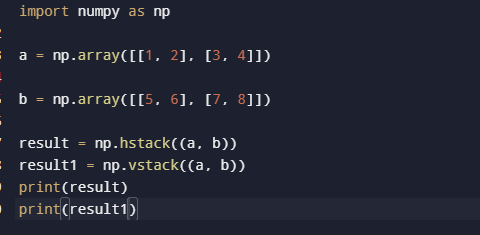


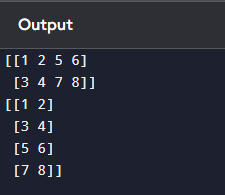




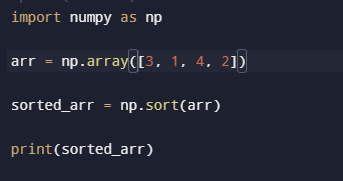
**Q.A Concatenate Two NumPy Arrays Horizontally and Vertically**  
  
● **Library Import**: NumPy is imported as np to access array functions.  
● **Array Creation**: Two 2D arrays a and b are created with shape (2, 2).  
● **Horizontal Concatenation**: np.hstack((a, b)) joins both arrays side-by-side (column-wise)

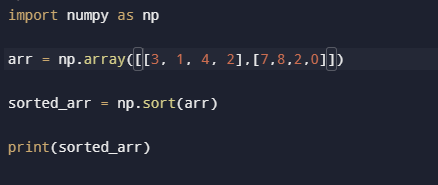
* **Vertical Concatenation**: np.vstack((a, b)) stacks arrays on top of each other (row-wise).  
  ● **Result Storage**: The result is stored in result, forming a shape of (2, 4).  
  ● **Output Display**: print(result) shows the final concatenated array.

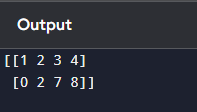




**Q.A Sort a NumPy Array in Ascending Order?**  
  
● **Library Import**: numpy is imported as np to use NumPy functions.  
● **Array Creation**: A 1D NumPy array arr is created with unsorted elements.  
● **Sorting Function**: np.sort(arr) sorts the array in ascending order.  
● **Returns Copy**: The function returns a sorted copy of the array without changing the original.  
● **Store Result**: The sorted array is stored in the variable sorted\_arr.  
● **Display Output**: print(sorted\_arr) prints the sorted array: [1 2 3 4].







**Q.A Sort a NumPy Array in Descending Order?**  
  
● **Library Import**: numpy is imported as np to access NumPy functions.  
● **Array Creation**: A 1D NumPy array arr is created with unordered numbers.  
● **Sorting in Ascending Order**: np.sort(arr) first sorts the array in ascending order: [1 2 3 4].  
● **Reverse the Array**: [::-1] is used to reverse the sorted array, converting it to descending order.  
● **Store Result**: The reversed sorted array is stored in sorted\_arr.  
● **Display Output**: print(sorted\_arr) prints the final descending array: [4 3 2 1].

