```
Start coding or generate with AI.
```

```
import numpy as np
arr = np.arange(1, 11)
print(arr)
print(type(arr))
```

```
[ 1  2  3  4  5  6  7  8  9 10]
<class 'numpy.ndarray'>
```

Create a 3x3 NumPy array filled with zeros.

```
arr = np.zeros((3, 3))
print(arr)
```

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

Create a 4x4 identity matrix using NumPy.

```
arr = np.eye(4)
print(arr)
```

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

Generate a 5x5 array filled with random numbers between 0 and 1.

```
arr = np.random.rand(5, 5)
print(arr)
```

```
[[0.74139522 0.32134786 0.83513615 0.09549294 0.71646374]
 [0.04391338 0.25528299 0.04984405 0.81295337 0.24426791]
 [0.01796099 0.88810347 0.27242051 0.23176107 0.41495876]
 [0.44144835 0.67939488 0.99433775 0.72221392 0.6607667 ]
 [0.62281504 0.69719221 0.37023992 0.14892655 0.91836712]]
```

Create a NumPy array of 10 elements, all initialized to the value 7.

```
arr = np.full(10, 7)
print(arr)
```

```
[7 7 7 7 7 7 7 7 7 7]
```

Reshape a 1D array of size 12 into a 3x4 2D array.

```
arr = np.arange(1, 13).reshape(3, 4)
print(arr)
```

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

Find the dimensions, shape, and size of a given NumPy array.

```
arr = np.array([[1, 2], [3, 4], [5, 6]])
print("Dimensions:", arr.ndim)
print("Shape:", arr.shape)
print("Size:", arr.size)
```

```
Dimensions: 2
Shape: (3, 2)
Size: 6
```

Extract all even numbers from a NumPy array [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].

```
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
evens = arr[arr % 2 == 0]
print(evens)
print(type(evens))
```

```
[ 2  4  6  8 10]
<class 'numpy.ndarray'>
```

Indexing and Slicing

```
 Access the second row and third column of a 2D array.
```

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

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
6
```

Slice a 3x3 array to extract the top-left 2x2 sub-array.

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

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
[[1 2]
 [4 5]]
```

Replace all negative elements in an array with zero.

```
arr = np.array([-1, 2, -3, 4, -5])
arr[arr < 0] = 0
print(arr)
```

```
[0 2 0 4 0]
```

Reverse the rows of a given 2D array.

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

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
[[7 8 9]
 [4 5 6]
 [1 2 3]]
```

Retrieve all elements in an array greater than a given value kk.

```
k = 5
arr = np.array([1, 6, 3, 9, 7])
print(arr[arr > k])
```

```
[6 9 7]
```

Array Operations

```
 Perform element-wise addition, subtraction, multiplication, and division of two arrays.
```

```
arr1 = np.array([1, 2, 3])
print(arr1)
print(arr1.shape)
```

```
arr2 = np.array([4, 5, 6])
print("Addition:", arr1 + arr2)
print("Subtraction:", arr1 - arr2)
print("Multiplication:", arr1 * arr2)
print("Division:", arr1 / arr2)
```

```
[1 2 3]
(3,)
Addition: [5 7 9]
Subtraction: [-3 -3 -3]
Multiplication: [ 4 10 18]
Division: [0.25 0.4  0.5 ]
```

Compute the dot product of two 2D arrays.

```
arr1 = np.array([[1, 2], [3, 4]])
arr2 = np.array([[5, 6], [7, 8]])
print(np.dot(arr1, arr2))
```

```
[[19 22]
 [43 50]]
```

Normalize an array so that its values lie between 0 and 1.

```
arr = np.array([10, 20, 30, 40, 50])
normalized = (arr - arr.min()) / (arr.max() - arr.min())
print(normalized)
```

```
[0.   0.25 0.5  0.75 1.  ]
```

Calculate the cumulative sum of elements in a NumPy array.

```
arr = np.array([1, 2, 3, 4])
print(np.cumsum(arr))
```

```
[ 1  3  6 10]
```

Compute the determinant and inverse of a 2x2 matrix.

```
from numpy.linalg import det, inv
arr = np.array([[1, 2], [3, 4]])
print("Determinant:", det(arr))
print("Inverse:", inv(arr))
```

```
Determinant: -2.0000000000000004
Inverse: [[-2.   1. ]
 [ 1.5 -0.5]]
```

Start coding or generate with AI.

Statistical and Mathematical Functions

```
Calculate the mean, median, standard deviation, and variance of a NumPy array.
```

```
arr = np.array([1, 2, 3, 4, 5])
print("Mean:", np.mean(arr))
print("Median:", np.median(arr))
print("Standard Deviation:", np.std(arr))
print("Variance:", np.var(arr))
```

Find the index of the maximum and minimum values in a NumPy array.

```
arr = np.array([10, 20, 5, 30])
print("Index of max:", np.argmax(arr))
print("Index of min:", np.argmin(arr))
```

Sort a NumPy array in ascending and descending order.

```
arr = np.array([5, 2, 9, 1])
print("Ascending:", np.sort(arr))
print("Descending:", np.sort(arr)[::-1])
```

Generate 10 random integers between 50 and 100 and find their sum.

```
arr = np.random.randint(50, 101, 10)
print("Random Integers:", arr)
print("Sum:", np.sum(arr))
```

Double-click (or enter) to edit

Compute the row-wise and column-wise sum of a 3x3 matrix.

```
arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print("Row-wise sum:", np.sum(arr, axis=1))
print("Column-wise sum:", np.sum(arr, axis=0))
```

```
    Row-wise sum: [ 6 15 24]
    Column-wise sum: [12 15 18]
```

Start coding or generate with AI.