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

```python
In [ ]:  a = np.array([1, 2, 3])
         print("Array a:", a)
```

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

```python
In [ ]:  b = np.arange(0, 10, 2)
         print("Array b:", b)
```

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

```python
In [ ]:  d = np.zeros((2, 3))
         print("Array d:\n", d)
```

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

```python
In [ ]:  e = np.ones((3, 2))
         print("Array e:\n", e)
```

```
Array e:
 [[1. 1.]
 [1. 1.]
 [1. 1.]]
```

```python
In [ ]:  f = np.eye(4)
         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.]]
```

```python
In [ ]:  a1 = np.array([1, 2, 3])
         reshaped = np.reshape(a1, (1, 3))
         print("Reshaped array:", reshaped)
```

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

```python
In [ ]:  f1 = np.array([[1, 2], [3, 4]])
         flattened = np.ravel(f1)
         print("Flattened array:", flattened)
```

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

```python
In [ ]:  e1 = np.array([[1, 2], [3, 4]])
         transposed = np.transpose(e1)
         print("Transposed array:\n", transposed)
```

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

```python
In [ ]:  a2 = np.array([1, 2])
         b2 = np.array([3, 4])
         stacked = np.vstack([a2, b2])
         print("Stacked arrays:\n", stacked)
```

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

In [ ]:
```python
g = np.array([1, 2, 3, 4])
added = np.add(g, 2)
print("Added 2 to g:", added)
```

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

In [ ]:
```python
squared = np.power(g, 2)
print("Squared g:", squared)
```

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

In [ ]:
```python
sqrt_val = np.sqrt(g)
print("Square root of g:", sqrt_val)
```

```
Square root of g: [1.         1.41421356 1.73205081 2.        ]
```

In [14]:
```python
print(a1)
print(g)
```

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

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

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[15], 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)
```

In [ ]:
```python
print(a)
print(a1)
```

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

In [ ]:
```python
a3 = np.array([1, 2, 3])
dot_product = np.dot(a1, a)
print("Dot product of a1 and a:", dot_product)
```

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

In [17]:
```python
s = np.array([1, 2, 3, 4])
mean = np.mean(s)
print("Mean of s:", mean)
```

```
Mean of s: 2.5
```

In [ ]:
```python
std_dev = np.std(s)
print("Standard deviation of s:", std_dev)
```

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

```python
minimum = np.min(s)
print("Min of s:", minimum)
```

Min of s: 1

```python
maximum = np.max(s)
print("Max of s:", maximum)
```

Max of s: 4

```python
matrix = np.array([[1, 2], [3, 4]])
```

```python
random_vals = np.random.rand(3)
print("Random values:", random_vals)
```

Random values: [0.50467973 0.1184663 0.76854211]

```python
np.random.seed(0)

random_vals = np.random.rand(3)
print("Random values:", random_vals)
```

Random values: [0.5488135 0.71518937 0.60276338]

```python
rand_ints = np.random.randint(0, 10, size=5)
print("Random integers:", rand_ints)
```

Random integers: [3 7 9 3 5]

```python
np.random.seed(0)

rand_ints = np.random.randint(0, 10, size=5)
print("Random integers:", rand_ints)
```

Random integers: [5 0 3 3 7]

```python
logical_test = np.array([True, False, True])
all_true = np.all(logical_test)
print("All elements True:", all_true)
```

All elements True: False

```python
logical_test = np.array([True, False, True])
all_true = np.all(logical_test)
print("All elements True:", all_true)
```

All elements True: False

```python
logical_test = np.array([False, False, False])
all_true = np.all(logical_test)
print("All elements True:", all_true)
```

All elements True: False

```python
any_true = np.any(logical_test)
print("Any elements True:", any_true)
```

Any elements True: False

```python
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]
```

```
In [ ]: union = np.union1d(set_a, set_b)
        print("Union of a and b:", union)
```

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

```
In [ ]: a = np.array([1, 2, 3])
        shape = a.shape
        size = a.size
        dimensions = a.ndim
        dtype = a.dtype

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

```
Shape of a: (3,)
Size of a: 3
Number of dimensions of a: 1
Data type of a: int64
```

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

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

```
In [ ]: array_size_in_bytes = a.nbytes
        print("Size of a in bytes:", array_size_in_bytes)
```

```
Size of a in bytes: 24
```

```
In [ ]: shared = np.shares_memory(a, copied_array)
        print("Do a and copied_array share memory?", shared)
```

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