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

In [1]:
```
print('21052410')
import numpy as np

null_vector = np.zeros(10)
null_vector[4] = 1

print(null_vector)
```

```
21052410
[0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
```

2. Create a vector with values ranging from 10 to 49

In [2]:
```
print('21052410')
import numpy as np

vector = np.arange(10, 50)

print(vector)
```

```
21052410
[10 11 12 13 14 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 45 46 47 48 49]
```

3. Reverse a vector (first element becomes last)

In [3]:
```
print('21052410')
vector = [1, 2, 3, 4, 5]
reversed_vector = vector[::-1]

print(reversed_vector)
```

```
21052410
[5, 4, 3, 2, 1]
```

4. Create a 3x3 matrix with values ranging from 0 to 8 `hint: reshape`

In [4]:
```
print('21052410')
import numpy as np

values = np.arange(9)

matrix = np.reshape(values, (3, 3))

print(matrix)
```

```
21052410
[[0 1 2]
 [3 4 5]
 [6 7 8]]
```

5. Find indices of non-zero elements from [1,2,0,0,4,0] `hint: np.nonzero`

In [7]:
```
print('21052410')
import numpy as np

arr = np.array([1, 2, 0, 0, 4, 0])

indices_nonzero = np.nonzero(arr)

print(indices_nonzero)
```

```
21052410
(array([0, 1, 4], dtype=int64),)
```

6. Create a 3x3x3 array with random values `hint: np.random.random`

In [12]:
```
print('21052410')
import numpy as np

array = np.random.random((3, 3, 3))

print(array)
```

```
21052410
[[[0.48013795 0.57209897 0.412109  ]
  [0.12540653 0.36104989 0.76567448]
  [0.11519156 0.32645557 0.24045984]]

 [[0.63595673 0.28808328 0.73144698]
  [0.29637435 0.93800564 0.91069837]
  [0.50325916 0.53306617 0.30038316]]

 [[0.0414752  0.84280816 0.46883511]
  [0.84393628 0.08218912 0.1253726 ]
  [0.06480153 0.72864659 0.80716669]]]
```

7. Create a 10x10 array with random values and find the minimum and maximum values
`hint: min, max`

In [34]:
```
print('21052410')
import numpy as np

array = np.random.randint(0, 101, size=(10, 10))

min_value = np.min(array)
max_value = np.max(array)

print("Minimum value:", min_value)
print("Maximum value:", max_value)
```

```
21052410
Minimum value: 0
Maximum value: 99
```

8. Create a random vector of size 30 and find the mean value `hint: mean`

In [17]:
```
print('21052410')
import numpy as np

vector = np.random.random(30)

mean_value = np.mean(vector)

print("Mean value:", mean_value)
```

```
21052410
Mean value: 0.5176277516930122
```

9. Create a 2d array with 1 on the border and 0 inside `hint: array[1:-1, 1:-1]`

In [26]:
```
print('21052410')
import numpy as np

array = np.ones((5, 5), dtype=int)

array[1:-1, 1:-1] = 0  # Set the inner part to 0
print(array)
```

```
21052410
[[1 1 1 1 1]
 [1 0 0 0 1]
 [1 0 0 0 1]
 [1 0 0 0 1]
 [1 1 1 1 1]]
```

10. Normalize a 5x5 random matrix `hint: (x -mean)/std`

In [33]:
```
print('21052410')
import numpy as np

matrix = np.random.randint(0, 10, size=(5, 5))

mean = np.mean(matrix)
std = np.std(matrix)

normalized_matrix = (matrix - mean) / std

print("Original Matrix:")
print(matrix)
print("\nNormalized Matrix:")
print(np.round(normalized_matrix,2))
```

```
21052410
Original Matrix:
[[1 9 3 3 1]
 [2 8 8 4 3]
 [2 1 7 0 1]
 [5 5 4 0 6]
 [5 9 7 6 6]]

Normalized Matrix:
[[-1.17  1.72 -0.45 -0.45 -1.17]
 [-0.81  1.35  1.35 -0.09 -0.45]
 [-0.81 -1.17  0.99 -1.53 -1.17]
 [ 0.27  0.27 -0.09 -1.53  0.63]
 [ 0.27  1.72  0.99  0.63  0.63]]
```

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

In [37]:
```
print('21052410')
import numpy as np

matrix_a = np.random.randint(0, 10, size=(5, 3))

matrix_b = np.random.randint(0, 10, size=(3, 2))

result = np.dot(matrix_a, matrix_b)

# result = matrix_a @ matrix_b
print("Matrix 1 :")
print(matrix_a)
print("Matrix 2 :")
print(matrix_b)
print("Resulting Matrix:")
print(result)
```

```
21052410
Matrix 1 :
[[6 4 8]
 [3 4 8]
 [0 1 8]
 [0 1 2]
 [1 6 1]]
Matrix 2 :
[[4 9]
 [0 4]
 [1 2]]
Resulting Matrix:
[[32 86]
 [20 59]
 [ 8 20]
 [ 2  8]
 [ 5 35]]
```

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

In [38]:
```
print('21052410')
import numpy as np

arr = np.array([1, 5, 7, 2, 9])
arr[(arr >= 3) & (arr <= 8)] *= 0
print(arr[arr!=0])
```

```
21052410
[1 2 9]
```

13. Find the eigenvalues and eigenvectors of a square matrix. `hint: np.linalg.eig`

In [45]:
```
print('21052410')
import numpy as np

n = int(input('Enter square matrix size'))
matrix = np.random.randint(0, 10, size=(n,n))

eigenvalues, eigenvectors = np.linalg.eig(matrix)
print('Matrix:')
print(matrix)
print("Eigenvalues:", eigenvalues)
print("Eigenvectors:")
print(np.round(eigenvectors,3))
```

```
21052410
Enter square matrix size3
Matrix:
[[3 7 7]
 [7 9 0]
 [9 0 4]]
Eigenvalues: [-6.31060765 15.69923518  6.61137247]
Eigenvectors:
[[ 0.712 -0.61   0.216]
 [-0.326 -0.638 -0.633]
 [-0.622 -0.47   0.744]]
```

14. Find the inverse of a square matrix. `hint: np.linalg.inv`

In [46]:
```
print('21052410')
import numpy as np

n = int(input('Enter square matrix size'))
matrix = np.random.randint(0, 10, size=(n,n))

inverse_matrix = np.linalg.inv(matrix)
print("Original Matrix:")
print(matrix)
print("\nInverse Matrix:")
print(inverse_matrix)
```

```
21052410
Enter square matrix size3
Original Matrix:
[[0 4 2]
 [3 8 0]
 [6 3 1]]

Inverse Matrix:
[[-0.08888889 -0.02222222  0.17777778]
 [ 0.03333333  0.13333333 -0.06666667]
 [ 0.43333333 -0.26666667  0.13333333]]
```