# 1

import numpy as np  
  
# Create a NumPy array with values 1 through 9  
a = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9])  
  
# Reshape the array to a 3x3 matrix  
a = a.reshape(3, 3)  
  
# Print the resulting matrix  
print(a)

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

# 2

import numpy as np  
  
b = np.array([5, 6, 7, 8, 9, 10, 11, 12, 13])  
b = b.reshape((3, 3))  
print(b)

[[ 5 6 7]  
 [ 8 9 10]  
 [11 12 13]]

# 3

import numpy as np  
  
a = np.array([[1,2,3],[4,5,6],[7,8,9]])  
b = np.array([[5,6,7],[8,9,10],[11,12,13]])  
  
result = a @ b  
  
print(result)

[[ 54 60 66]  
 [126 141 156]  
 [198 222 246]]

# 4

import numpy as np  
  
result = np.dot(a[0], b[:,0])  
print(result)

54

# 5

import numpy as np  
  
a = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])  
sum\_a = np.sum(a)  
print(sum\_a)

45

# 6

import numpy as np  
  
b = np.array([[5, 6, 7], [8, 9, 10], [11, 12, 13]])  
  
product = np.prod(b)  
print(product)

259459200

# 7

import numpy as np  
  
c = np.array([1, 2, 3, 4, 5])  
c = np.insert(c, 2, 0)  
  
print(c)

[1 2 0 3 4 5]

# 8

import numpy as np  
  
d = np.array([1, 2, 3, 4, 5])  
d = np.delete(d, 3)  
print(d)

[1 2 3 5]

# 9

import numpy as np  
  
e = np.arange(1, 13).reshape(3, 4)  
print("e:")  
print(e)  
  
greater\_than\_5 = e[e > 5]  
print("Elements greater than 5:")  
print(greater\_than\_5)

e:  
[[ 1 2 3 4]  
 [ 5 6 7 8]  
 [ 9 10 11 12]]  
Elements greater than 5:  
[ 6 7 8 9 10 11 12]

# 10

import numpy as np  
f = np.arange(1, 21).reshape(4, 5)  
selected = f[[0, 1, 2], [0, 2, 4]]  
  
print(selected)

[ 1 8 15]

# 11

import numpy as np  
  
a = np.zeros((5, 5))   
np.fill\_diagonal(a, 0.5)   
print(a)

[[0.5 0. 0. 0. 0. ]  
 [0. 0.5 0. 0. 0. ]  
 [0. 0. 0.5 0. 0. ]  
 [0. 0. 0. 0.5 0. ]  
 [0. 0. 0. 0. 0.5]]

# 12

import numpy as np  
  
b = np.random.randint(0, 11, size=(3, 3, 3))  
print(b)

[[[ 0 5 10]  
 [ 9 9 7]  
 [ 0 8 10]]  
  
 [[ 7 0 8]  
 [ 1 3 9]  
 [ 9 5 9]]  
  
 [[ 8 10 5]  
 [ 7 1 6]  
 [ 6 0 5]]]

# 13

import numpy as np  
b = np.random.randint(0, 10, (3, 3, 3))  
row\_means = np.mean(b, axis=2)  
  
print(row\_means)

[[6. 3.33333333 5. ]  
 [1.66666667 4.33333333 3.66666667]  
 [3.33333333 6. 2. ]]

# 14

import numpy as np  
  
a = np.eye(3)  
std\_dev = np.std(a, axis=0)  
  
print(std\_dev)

[0.47140452 0.47140452 0.47140452]

# 15

import numpy as np  
  
a = np.diag([0.5] \* 5)  
a @ a.T

array([[0.25, 0. , 0. , 0. , 0. ],  
 [0. , 0.25, 0. , 0. , 0. ],  
 [0. , 0. , 0.25, 0. , 0. ],  
 [0. , 0. , 0. , 0.25, 0. ],  
 [0. , 0. , 0. , 0. , 0.25]])

# 16

import numpy as np  
c = np.eye(6)  
c = np.roll(c, 1, axis=1)  
print(c)

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

# 17

import numpy as np  
d = np.random.rand(4, 4)  
row\_sums = d.sum(axis=1, keepdims=True)  
d\_normalized = d / row\_sums  
  
print(d\_normalized)

[[0.05189333 0.25988491 0.5815722 0.10664955]  
 [0.12406285 0.09954508 0.25819015 0.51820192]  
 [0.31527049 0.03958109 0.40104358 0.24410484]  
 [0.01727134 0.54384085 0.06707629 0.37181152]]

# 18

import numpy as np  
  
e = np.arange(1, 10).reshape(3, 3)  
print("Original matrix:\n", e)  
  
e\_flipped = np.flipud(e)  
print("Flipped matrix:\n", e\_flipped)

Original matrix:  
 [[1 2 3]  
 [4 5 6]  
 [7 8 9]]  
Flipped matrix:  
 [[7 8 9]  
 [4 5 6]  
 [1 2 3]]

# 19

import numpy as np  
f = np.random.randint(0, 10, (4,4))  
f.sort(axis=1)  
  
print(f)

[[1 2 2 6]  
 [0 4 4 9]  
 [1 5 5 6]  
 [0 0 1 4]]

# 20