1.How do you import the NumPy library using an alias?

import numpy as np

2.How do you check the version and configuration of NumPy?

```
import numpy as np
print(np.__version__)
```

```
2.0.2
```

3.How do you create a vector filled with zeros of size 10?

```
import numpy as np
vector=np.zeros(10)
print(vector)
```

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

4.How do you find help/documentation for a NumPy function from the command line?

```
np.info(np.zeros)

zeros(shape, dtype=float, order='C', *, like=None)

Return a new array of given shape and type, filled with zeros.

Parameters
----------
shape : int or tuple of ints
    Shape of the new array, e.g., ``(2, 3)`` or ``2``.
dtype : data-type, optional
    The desired data-type for the array, e.g., `numpy.int8`.  Default is
    `numpy.float64`.
order : {'C', 'F'}, optional, default: 'C'
    Whether to store multi-dimensional data in row-major
    (C-style) or column-major (Fortran-style) order in
    memory.
like : array_like, optional
    Reference object to allow the creation of arrays which are not
    NumPy arrays. If an array-like passed in as ``like`` supports
    the ``__array_function__`` protocol, the result will be defined
    by it. In this case, it ensures the creation of an array object
    compatible with that passed in via this argument.

    .. versionadded:: 1.20.0

Returns
-------
out : ndarray
    Array of zeros with the given shape, dtype, and order.

See Also
```

```
--------
zeros_like : Return an array of zeros with shape and type of input.
empty : Return a new uninitialized array.
ones : Return a new array setting values to one.
full : Return a new array of given shape filled with value.

Examples
--------
>>> np.zeros(5)
array([ 0.,  0.,  0.,  0.,  0.])

>>> np.zeros((5,), dtype=int)
array([0, 0, 0, 0, 0])

>>> np.zeros((2, 1))
array([[ 0.],
       [ 0.]])

>>> s = (2,2)
>>> np.zeros(s)
array([[ 0.,  0.],
       [ 0.,  0.]])

>>> np.zeros((2,), dtype=[('x', 'i4'), ('y', 'i4')]) # custom dtype
array([(0, 0), (0, 0)],
      dtype=[('x', '<i4'), ('y', '<i4')])
```

5.How do you create a zero vector of size 10 with the fifth element as 1?

```
a=np.zeros(10)
a[4]=1
print(a)
```

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

6.How do you create a vector with values from 10 to 49?

```
a=np.arange(10,49)
print(a)
```

```
[10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48]
```

7.How do you reverse a vector?

```
a=np.arange(10,49)
a[::-1]
```

```
array([48, 47, 46, 45, 44, 43, 42, 41, 40, 39, 38, 37, 36, 35, 34, 33,
32,
       31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16,
15,
       14, 13, 12, 11, 10])
```

8.How do you create a 3x3 matrix with values from 0 to 8?

```
np.arange(9).reshape((3,3))
```

```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

9.How do you find indices of non-zero elements in an array?

```
a=np.array([1,2,0,7,0,0])
print(np.nonzero(a))
```

```
(array([0, 1, 3]),)
```

10.How do you create a 3x3 identity matrix?

```
print(np.eye(3))
```

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

11.How do you generate a 3x3x3 array with random values?

```
import numpy as np
array=np.random.rand(3, 3, 3)
print(array)
```

```
[[[0.16011503 0.36355034 0.42981045]
  [0.06491562 0.0436084  0.74810146]
  [0.58051122 0.33998582 0.55050963]]

 [[0.2148906  0.83791132 0.99071484]
  [0.34939303 0.01625633 0.58803253]
  [0.0308488  0.51726989 0.26038145]]

 [[0.00256804 0.08009062 0.43421959]
  [0.11986726 0.49867192 0.82919677]
  [0.18156944 0.67211673 0.52137614]]]
```

12.How do you find the min and max values in a 10x10 random matrix?

```
import numpy as np
matrix=np.random.rand(10,10)
min_value=np.min(matrix)
max_value=np.max(matrix)
print("minimum value:",min_value)
print("maximum value:",max_value)
```

```
minimum value: 0.0030985522861305403
maximum value: 0.9800831600256107
```

13.How do you calculate the mean of a random vector of size 30?

```
import numpy as np
vector=np.random.rand(10)
mean_value=np.mean(vector)
print("Mean:",mean_value)
```

```
Mean: 0.4915716887792635
```

14.How do you create a 2D array with 1s on the border and 0s inside?

```
import numpy as np

array = np.ones((5, 5), dtype=int)  # Start with all 1s
array[1:-1, 1:-1] = 0               # Set the inner part to 0

print(array)
```

```
[[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]]
```

15.What is the result of operations like 0 * np.nan, np.nan == np.nan, etc.?

```
import numpy as np
result=0*np.nan
print(result)
```

```
nan
```

```
print(np.nan == np.nan)
```

```
False
```

16.How do you create a matrix with values below the diagonal?

```
import numpy as np

matrix = np.arange(1, 10).reshape(3, 3)
lower_triangle = np.tril(matrix)
```

```
print("Original Matrix:\n", matrix)
print("Lower Triangle:\n", lower_triangle)
```

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

17.How do you create a checkerboard pattern in an 8x8 matrix?

```
import numpy as np

checkerboard = np.zeros((8, 8), dtype=int)
checkerboard[1::2, ::2] = 1  # Set 1s on alternating rows and columns
checkerboard[::2, 1::2] = 1
print("checkerboard:")
print(checkerboard)
```

```
checkerboard:
[[0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]]
```

18.How do you find the index of a flat element in a 3D shape?

```
import numpy as np

# Example: a 3D array with shape (3, 4, 5)
shape = (3, 4, 5)
flat_index = 42

index_3d = np.unravel_index(flat_index, shape)
print(index_3d)
```

```
(np.int64(2), np.int64(0), np.int64(2))
```

20.How do you normalize a matrix (values between 0 and 1)?

```
import numpy as np

matrix = np.random.randint(0, 100, (3, 3))  # Example matrix
normalized = (matrix - np.min(matrix)) / (np.max(matrix) - np.min(matrix))

print("Original Matrix:\n", matrix)
print("Normalized Matrix:\n", normalized)
```

```
Original Matrix:
 [[20 31 55]
 [78 54 10]
 [ 9 14 38]]
Normalized Matrix:
 [[0.15942029 0.31884058 0.66666667]
 [1.         0.65217391 0.01449275]
 [0.         0.07246377 0.42028986]]
```

21.How do you define a custom data type for storing color (RGBA)?

```
import numpy as np

rgba_dtype = np.dtype([
    ('r', np.uint8),
    ('g', np.uint8),
    ('b', np.uint8),
    ('a', np.uint8)
])

colors = np.array([
    (255, 0, 0, 255),
    (0, 255, 0, 255),
    (0, 0, 255, 255)
], dtype=rgba_dtype)

print(colors)
```

```
[(255,   0,   0, 255) (  0, 255,   0, 255) (  0,   0, 255, 255)]
```

22.How do you perform a matrix multiplication between two matrices?

```
import numpy as np

# Create two sample matrices
matrix1 = np.array([[1, 2], [3, 4]])
matrix2 = np.array([[5, 6], [7, 8]])

result=matrix1@matrix2
print(result)
```

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

23.How do you negate values between two given numbers in a vector?

```
vector=np.array([1,2,3,4,7,9,3,11,13])
lower_bound=3
upper_bound=7
res = np.array([eval('-'+str(i)) for i in vector if i>=lower_bound and i<=up
print(res)
```

```
[-3 -4 -7 -3]
```

24.What is the output of sum(range(5), -1) vs np.sum(range(5), -1)?

```
import numpy as np
np.sum(range(5), -1)
```

```
np.int64(10)
```

25.Which NumPy vector operations are legal or illegal?

```
a=np.array([1,2,3])
b=np.array([4,5,6])
print(a+b)
print(a-b)
print(a*b)
print(a/b)
print(a**b)
```

```
[5 7 9]
[-3 -3 -3]
[ 4 10 18]
[0.25 0.4  0.5 ]
[  1  32 729]
```

26.What happens when you divide integers or floats by zero in NumPy?

```
a=np.array([1,2,3,4])
b=np.array([1.2,2.4,3.5,6.7])
print(a/0)
print(b/0)
```

```
[inf inf inf inf]
[inf inf inf inf]
/tmp/ipython-input-3523185438.py:3: RuntimeWarning: divide by zero enc
  print(a/0)
/tmp/ipython-input-3523185438.py:4: RuntimeWarning: divide by zero enc
  print(b/0)
```

27.How do you round a float array away from zero?

```
import numpy as np

x = np.array([1.2, -1.7, 2.5, -3.3, 0.0])

# Round away from zero
rounded = np.copysign(np.ceil(np.abs(x)), x)

print(rounded)
```

```
[ 2. -2.  3. -4.  0.]
```

29.How do you create a 5x5 matrix with row values ranging from 0 to 4?

```
import numpy as np

matrix = np.tile(np.arange(5), (5, 1))
print(matrix)
```

```
[[0 1 2 3 4]
 [0 1 2 3 4]
 [0 1 2 3 4]
 [0 1 2 3 4]
 [0 1 2 3 4]]
```

30.How do you use a generator to create a NumPy array?

```
import numpy as np

gen = (x**2 for x in range(10))

arr = np.fromiter(gen, dtype=int)

print(arr)
```

```
[ 0  1  4  9 16 25 36 49 64 81]
```

31.How do you create an array with evenly spaced values between 0 and 1 (excluded)?

```
import numpy as np

arr = np.linspace(0, 1, num=5, endpoint=False)
print(arr)
```

```
[0.  0.2 0.4 0.6 0.8]
```

32.How do you sort a random array?

```
import numpy as np

arr = np.random.rand(10)  # 1D array with random floats
sorted_arr = np.sort(arr)

print("Original:", arr)
print("Sorted:  ", sorted_arr)
```

```
Original: [0.03174351 0.194379   0.24584079 0.30205063 0.51641609 0.328
 0.69240139 0.40832489 0.62286091 0.26276785]
Sorted:   [0.03174351 0.194379   0.24584079 0.26276785 0.30205063 0.328
 0.40832489 0.51641609 0.62286091 0.69240139]
```

33.How do you sum an array faster than using np.sum?

```
import numexpr as ne
import numpy as np

arr = np.random.rand(1_000_000)
total = ne.evaluate("sum(arr)")
print(total)
```

```
499612.7554538166
```

34.How do you check if two arrays are equal?

```
import numpy as np

array1 = np.array([1, 2, 3, 4])
array2 = np.array([1, 2, 3, 4])

are_equal = np.array_equal(array1, array2)

print(are_equal)
```

```
True
```

35.How do you make a numpy array read only?

```
import numpy as np

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

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

36.How do you convert Cartesian coordinates to polar coordinates?

```
import numpy as np

x = np.array([1, 3, -2])
y = np.array([1, -3, 2])
r = np.sqrt(x**2 + y**2)
theta = np.arctan2(y, x)
print("r=",r)
print("theta=",theta)
```

```
r= [1.41421356 4.24264069 2.82842712]
theta= [ 0.78539816 -0.78539816  2.35619449]
```

37.How do you replace the maximum value in an array with zero?

```
import numpy as np

a = np.arange(10)
max_index = np.argmax(a)
a[max_index] = 0
print(a)
```

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

38.How do you create a structured array for (x, y) coordinate pairs?

```
import numpy as np
dtype=[('x',float),('y',float)]
arr=np.array([(1,2),(3,4)],dtype=dtype)
print(arr)
```

```
[(1., 2.) (3., 4.)]
```

39.How do you create a Cauchy matrix using two arrays?

```
import numpy as np

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

C = 1.0 / (x[:, np.newaxis] - y[np.newaxis, :])

print(C)
```

```
[[-0.33333333 -0.25       -0.2        ]
 [-0.5        -0.33333333 -0.25       ]
 [-1.         -0.5        -0.33333333]]
```

40.How do you find the range of values for each NumPy scalar type?

```
import numpy as np
print(np.iinfo(np.int8))
print(np.iinfo(np.int16))
```

```
Machine parameters for int8
---------------------------------------------------------------
min = -128
max = 127
---------------------------------------------------------------

Machine parameters for int16
---------------------------------------------------------------
min = -32768
max = 32767
---------------------------------------------------------------
```

41.How do you display all values in a large array?

```
import numpy as np
arr=np.arange(100)
print(arr)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22
 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70
 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94
 96 97 98 99]
```

42.How do you find the closest value to a given scalar in an array?

```
import numpy as np
arr=np.array([3,7,10,15,20])
scalar=12
closest_value=arr[np.abs(arr-scalar).argmin()]
print(closest_value)
```

```
10
```

43.How do you create a structured array with position and color fields?

```
dtype=[('position', [('x', float), ('y', float)]), ('color', [('r', 'U10'),
data=np.array([((1.0,2.0),'red'),((3.5,4.2),'blue'),((2.9,1.2),'green')],dty
print("positions:/n",data['position'])
print("colors:/n",data['color'])
```

```
positions:/n [(1. , 2. ) (3.5, 4.2) (2.9, 1.2)]
colors:/n [('red', 'red', 'red') ('blue', 'blue', 'blue')
 ('green', 'green', 'green')]
```

44.How do you compute pairwise distances between coordinates?

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

distances = np.sqrt(((points[:,None,:] - points[None,:,:])**2).sum(axis=-1)

print(distances)
```

```
[[0.         2.82842712 5.65685425]
 [2.82842712 0.         2.82842712]
 [5.65685425 2.82842712 0.        ]]
```

45.How do you cast a float array to integer type in place?

```
import numpy as np

arr = np.array([1.2, 2.5, 3.9])
arr = arr.astype(int)

print(arr)
```

```
[1 2 3]
```

46.How do you read data from a file with missing values?

```
import numpy as np
arr=np.array([1.2,3.4,5.6,7.8])
arr=arr.astype(int)
print(arr)
```

```
[1 3 5 7]
```

47.How do you iterate through all elements in a NumPy array with indices? import numpy as np

```
a=np.array([1,2,3,4,5,6,7,8,9])
for index,value in np.ndenumerate(a):
    print(index,value)
```

```
(0,) 1
(1,) 2
(2,) 3
(3,) 4
(4,) 5
(5,) 6
(6,) 7
(7,) 8
(8,) 9
```

49.How do you randomly place a specific number of elements in a 2D array?

```
import numpy as np
a=np.array([(1,2),(3,4)])
a[0,1]
```

```
np.int64(2)
```

50.How do you subtract the row mean from a matrix?

```
import numpy as np
a=np.array([(1,2),(3,4)])
row_mean=np.mean(a,axis=1)
```

```
print(row_mean)
print(a-row_mean)
```

```
[1.5 3.5]
[[-0.5 -1.5]
 [ 1.5  0.5]]
```

51.How do you sort a 2D array based on the values of one column?

```
import numpy as np

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

sorted_array = a[a[:, 0].argsort()]

print(sorted_array)
```

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

52.How do you check if a 2D array contains any null (zero-only) columns?

```
import numpy as np
a=np.array([[0,1,2],
            [0,2,0],
            [0,3,0]])
zero_columns=np.all(a==0,axis=0)
print(zero_columns)
```

```
[ True False False]
```

53.How do you find the nearest value to a given number in a NumPy array?

```
import numpy as np
arr=np.array([1.2,2.3,4.5,5.2,6.7])
given_number=2.5
nearest_value=arr[np.abs(arr-given_number).argmin()]
print(nearest_value)
```

```
2.3
```

55.How do you increment elements in an array using an index array with repeats?

```
import numpy as np
arr=np.zeros(5,dtype=int)
indices=np.array([0,1,2,3,4])
np.add.at(arr,indices,1)
print(arr)
```

```
[1 1 1 1 1]
```

56.How do you accumulate values into an array based on indices?

```
import numpy as np
values=np.array([5,4,3,2,1])
indices=np.array([4,0,1,2,3])
arr=np.zeros(5,dtype=int)
np.add.at(arr,indices,values)
print(arr)
```

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

57.How do you count unique colors in a 3D image array?

```
import numpy as np
image=np.array([[[1,2,3],[4,5,6]],[[7,8,9],[10,11,12]]])
unique_colors=np.unique(image)
print(unique_colors)
```

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

58.How do you compute the sum across the last two axes of a 4D array? import numpy as np

```
a=np.array([[[[1,2,3],[4,5,6]],[[7,8,9],[10,11,12]]]])

print(a.sum(axis=(2,3)))
```

```
[[21 57]]
```

59.How do you calculate means of grouped values in an array?

```
#59.How do you calculate means of grouped values in an array?
groups=np.array([0,1,1,0,2,2,1])
values=np.array([10,20,25,30,5,15,35])
sums=np.bincount(groups,weights=values)
counts=np.bincount(groups)
means=sums/counts
print(means)
```

```
[20.         26.66666667 10.        ]
```

60.How do you efficiently get the diagonal of a matrix dot product?

```
import numpy as np
a=np.array([[1,2],[3,4]])
b=np.array([[5,6],[7,8]])
product=np.dot(a,b)
```

```
diag=np.diagonal(product)
print(diag)
```

```
[19 50]
```

62.How do you multiply a (5,5,3) array with a (5,5) array element-wise?

```
a=np.random.rand(5,5,3)
b=np.random.rand(5,5)
result=(a*b[:,:,np.newaxis])
print(result.shape)
```

```
(5, 5, 3)
```

63.How do you swap two rows in a matrix?

```
import numpy as np
a=np.array([[1,2,3],[4,5,6],[7,8,9]])
a[[0,1]]=a[[1,0]]
print(a)
```

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

64.How do you extract unique line segments from triangle definitions?

```
import numpy as np
triangles=np.array([[0,1,2],
                    [2,3,0]])
edges=np.concatenate([triangles[:,[0,1]],
                      triangles[:,[1,2]],
                      triangles[:,[2,0]]])
edges=np.sort(edges,axis=1)
unique_edges=np.unique(edges,axis=0)
print(unique_edges)
```

```
[[0 1]
 [0 2]
 [0 3]
 [1 2]
 [2 3]]
```

67.How do you create a rolling 2D view from a 1D array?

```
import numpy as np

arr = np.arange(10)
window_size = 3
itemsize = arr.itemsize
stride = arr.strides[0]
```

```
rolling_view = np.lib.stride_tricks.as_strided(arr, shape=(len(arr) - window
print(rolling_view)
```

```
[[0 1 2]
 [1 2 3]
 [2 3 4]
 [3 4 5]
 [4 5 6]
 [5 6 7]
 [6 7 8]
 [7 8 9]]
```

68.How do you invert booleans or negate floats in-place?

```
import numpy as np
x=np.array([False,True,True,False])
x[:]=~x
print(x)
```

```
[ True False False  True]
```

71.How do you extract a fixed-shaped subarray centered on a given point?

```
import numpy as np
a=np.arange(25).reshape(5,5)
center_point=(1,2)
half_size=1
r,c=center_point
subarray=a[r-half_size:r+half_size+1,c-half_size:c+half_size+1]
print(subarray)
```

```
[[ 1  2  3]
 [ 6  7  8]
 [11 12 13]]
```

72.How do you generate overlapping subarrays from a vector? import numpy as np

```
import numpy as np
arr = np.arange(10)
subarray_size = 3
step_size = 1
itemsize = arr.itemsize
stride = arr.strides[0]
rolling_view = np.lib.stride_tricks.as_strided(arr, shape=((len(arr) - subar
print(rolling_view)
```

```
[[0 1 2]
 [1 2 3]
 [2 3 4]
 [3 4 5]
 [4 5 6]
 [5 6 7]
 [6 7 8]
 [7 8 9]]
```

74.How do you find the most frequent value in an array?

```
import numpy as np

arr = np.array([1, 2, 3, 2, 1, 2, 3, 4, 5, 4, 4, 4])
counts = np.bincount(arr)
most_frequent_value = np.argmax(counts)
print("Most frequent value:", most_frequent_value)
```

```
Most frequent value: 4
```

76.How do you create a 2D symmetric matrix class?

```
a=np.random.randint(1,10,(4,4))
symmetric_matrix=(a+a.T)/2
print(symmetric_matrix)
```

```
[[2.  7.  3.5 5. ]
 [7.  1.  2.  6.5]
 [3.5 2.  1.  3. ]
 [5.  6.5 3.  4. ]]
```

77.How do you compute the sum of multiple matrix-vector products at once?

```
import numpy as np
matrices=np.random.randint(1,10,(3,3,3))
vectors=np.random.randint(1,10,(3,3))
result=np.zeros(3)
for A,x in zip(matrices,vectors):
    result+=A @ x
print(result)
```

```
[263. 234. 236.]
```

80.How do you extract the top N largest values from an array?

```
import numpy as np
arr=np.array([10,5,8,12,3,7])
n=3
largest_values=arr[np.argsort(arr)[-n:]]
print(largest_values)
```

```
[ 8 10 12]
```

81.How do you create a Cartesian product of multiple input arrays?

```
import numpy as np
from itertools import product

a=np.array([1,2])
```

```
b=np.array([3,4])
c=np.array([5,6])
cartesian_product=np.array(list(product(a,b,c)))
print(cartesian_product)
```

```
[[1 3 5]
 [1 3 6]
 [1 4 5]
 [1 4 6]
 [2 3 5]
 [2 3 6]
 [2 4 5]
 [2 4 6]]
```

82.How do you create a record array from a regular NumPy array?

```
import numpy as np

regular_array = np.array([(1, 'Alice', 25),
                          (2, 'Bob', 30),
                          (3, 'Charlie', 22)],
                         dtype=[('id', int), ('name', 'U10'), ('age', int)]
record_array = np.rec.array(regular_array)
print(record_array)
```

```
[(1, 'Alice', 25) (2, 'Bob', 30) (3, 'Charlie', 22)]
```

84.How do you find rows in a matrix containing all elements of another matrix?

```
import numpy as np
A=np.array([[1,2,3,4],
            [4,5,6,7],
            [2,3,4,5],
            [7,8,9,1]])
B=np.array([2,3,4,5])
contains_elements = np.isin(A, B)
rows_containing_all_elements = np.all(contains_elements, axis=1)
result_rows = A[rows_containing_all_elements]
print(result_rows)
```

```
[[2 3 4 5]]
```

86.How do you convert integers to their binary matrix representation?

```
import numpy as np
arr=np.array([1,2,3],dtype=np.uint8)
binary_matrix=np.unpackbits(arr[:,np.newaxis],axis=1)
print(binary_matrix)
```

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

87.How do you extract unique rows from a 2D array?

```
a=np.array([[1,2,3],
            [1,2,3],
            [4,5,6],
            [1,2,4]])
unique_rows=np.unique(a,axis=0)
print(unique_rows)
```

```
[[1 2 3]
 [1 2 4]
 [4 5 6]]
```

90.How do you filter 2D array rows that sum to a specific number using only integers?

```
import numpy as np
arr=np.array([[1,2,3],
              [4,1,1],
              [2,2,2],
              [3,3,0]],dtype=int)
specific_sum=6
rows_sum=np.sum(arr,axis=1)
filtered_rows=arr[rows_sum==specific_sum]
print(filtered_rows)
```

```
[[1 2 3]
 [4 1 1]
 [2 2 2]
 [3 3 0]]
```