```
# 1
pip install numpy
```

```
# 2
import numpy as np
```

```
# 3
np_arr = np.array([1,2,3,4,5])
print(np_arr)
```

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

```
# 5
import numpy as np
arr1 = np.array([[[1,2,3],[4,5,6]],[[7,8,9],[10,11,12]]])
print(arr1)
print(arr1.ndim)
```

```
# 6
import numpy as np
np_arr = np.array([1,2,3,4],ndmin=3)
print(np_arr)
```

```
# 7
import numpy as np

np_arr = np.array([[1,2],[3,4]])
print(np_arr)
print(np_arr.ndim)
```

```
# 8
import numpy as np

np_zeros = np.zeros((2,3))
print(np_zeros)
```

```
# 9
import numpy as np

np_ones = np.ones((3,4))
print(np_ones)
```

```
# 10
import numpy as np

np_empty = np.empty(2)
print(np_empty)
```

```
# 11
import numpy as np

np_arange = np.arange(5)
#np_arange = np.arange(2,9,2)
print(np_arange)
```

```
# 12
import numpy as np
np_ls = np.linspace(2,10,5)
print(np_ls)
```

```
# 13
import numpy as np
print(np.array([[1,2,3],[4,5,6]]).shape)
```

```
# 14
import numpy as np
arr = np.array([[1,2,3,4],[4,5,6,7]])
print(arr.size)
```

```
# 15
import numpy as np
arr = np.array([[1,2,3,4],[4,5,6,7]])
print(f"Dimension: {arr.ndim}D")


# 16
import numpy as np
#arr = np.array([1,2,3], dtype = np.float64)
arr = np.array([1,2,3]).astype(np.float64)
print(arr.dtype)
```

```
     float64
```

```
# 17
import numpy as np
#arr = np.array([1,2,3]).astype(int)
arr1 = np.array([1,2,3], dtype=int)
print(arr.dtype)
print(arr)



# 18
import numpy as np

arr = np.array([3,7,1,8,4,6,0,2,5])
print("Max : ",max(arr)) # print(np.max(arr))
print("Min: ",min(arr))


# 19
import numpy as np

arr =np.arange(1,7)
print(arr.reshape(2,3))


# 20
import numpy as np

arr =np.arange(1,7)
split_arr = np.split(arr,3) #np.array_split(arr,4) -- unequal split
print(split_arr)

#arr1 = np.split(np.arange(2,8),3)


# 21
import numpy as np

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

#hsplit = np.hsplit(arr, 3)
hsplit = np.split(arr,3,axis=1)
print("\n","Horizontal Split: \n",hsplit)

#vsplit = np.vsplit(arr,3)
vsplit = np.split(arr,3,axis = 0)
print("\n","Vertical Split: \n",vsplit)



# 22
import numpy as np

arr = np.array([1,2,3,4])
new_dim = np.expand_dims(arr,axis = 0)
#new_dim = arr[:, np.newaxis]  # or arr[:, None] or arr[np.newaxis,:] for axis = 0
print(arr,arr.ndim)
print(new_dim,new_dim.ndim)


# 23
import numpy as np
arr = np.array([[1,2,3,4],[5,6,7,8]])
flat_arr = arr.flatten()
print(flat_arr)
```

```
# 24
import numpy as np
arr = np.array([9,1,5,8,4,2,7,3,6])
arr.sort()
#sorted = np.sort(arr)[::-1] # slicing for decending!!
print(arr)


# 25
import numpy as np
arr1 = np.array([[1,2],[3,4]])
arr2 = np.array([[5,6],[7,8]])
concat_arr = np.concatenate((arr1,arr2))
print(concat_arr)


# 26
import numpy as np

arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
result = np.add(arr1,arr2)
print(result)



# 27
import numpy as np

arr1 = np.array([1, 2, 3])
result = np.exp(arr1)
print(result)


# 28

import numpy as np

arr1 = np.array([1, 2, 3, 4,1])
print("sum:",np.sum(arr1))
print("mean:",np.mean(arr1))
print("median:",np.median(arr1))


# 29
import numpy as np

matrix1 = np.array([[1, 2, 3],
                    [4, 5, 6]])
matrix2 = np.array([[7, 8],
                    [9, 10],
                    [11, 12]])
result_dot = np.dot(matrix1, matrix2)
result_operator = matrix1 @ matrix2

print("Matrix 1:",matrix1)
print("\nMatrix 2:",matrix2)
print("\nResult (Matrix multiplication using np.dot()):",result_dot)
print("\nResult (Matrix multiplication using @ operator):",result_operator)



# 30
import numpy as np

matrix1 = np.array([[1, 2, 3],
                    [4, 5, 6]])
transposed = np.transpose(matrix1)
print(matrix1)
print("Transposed Matrix:\n",transposed)


# 31

'''
np.dot -- Used for matrix Multiplication

'''
```

```
# 32
import numpy as np
matrix = np.array([[1, 2],
                   [3, 4]])
inverse_matrix = np.linalg.inv(matrix) # pseudo_inverse_matrix = np.linalg.pinv(matrix)
print("Original Matrix:",matrix)
print("\nInverse Matrix:\n",inverse_matrix)


# 33
import numpy as np

matrix1 = np.array([[1, 2],
                    [3, 4]])
matrix2 = np.array([[5, 6],
                    [7, 8]])

matrix_product = np.matmul(matrix1, matrix2) #np.dot or @
print(matrix_product)



# 34

# NA


# 35
import numpy as np

arr = np.array([1, 2, 3, 4, 5])
indices = np.where(arr > 2)
print(indices)


# 36
import numpy as np
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
result = arr1 * arr2           # np.dot or @ returns cummulative product
#res_mul = np.multiply(arr1,arr2)
print(result)


# 37
import numpy as np
arr = np.array([[1, 2, 3],
                [4, 5, 9]])
mean_row = np.mean(arr, axis=1) # if axis = 0, it cals over columns
median_row = np.median(arr, axis=1)
print("mean of each row:",mean_row)
print("Median of each row:",median_row)



# 38
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
std_deviation = np.std(arr)
print("Standard deviation:", std_deviation)



# 40
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9])
sub_arr = arr[2:5]
print("Sub-array:", sub_arr)


# 41
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
index = np.where(arr == 3)[0]
#index = np.argwhere(arr == 3)
print("Index of value 3:", index)


# 42
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9])
sub_arr = arr[:5]
print("Sub-array:", sub_arr)
```

```
# 43
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
n = int(input("Enter nth element? "))
selected_elements = arr[n-1::n]
print("Selected elements:", selected_elements)



# 44
import numpy as np
arr = np.array([[1, 2, 3],
                    [4, 5, 6]])
specific_column = arr[:, 1]   # arr[0,:]  returns row - 0
print("Specific column:", specific_column)


# 45
import numpy as np
arr = np.array([[1, 2, 3],
                    [4, 5, 6]])

reversed_rows = arr[:, ::-1]
print(reversed_rows)
#new_arr = arr[:,][::-1]  -- this return only a specified row
#print("reversed row:", new_arr)


# 46
import numpy as np
arr = np.array([[1, 2, 3],
                [4, 5, 6],
                [7, 8, 9]])
diagonal = np.diagonal(arr)
print(diagonal)


# 47
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
boolean_array = arr > 2
sliced_array = arr[boolean_array]
print("Original array:", arr)
print("Boolean array based on condition (arr > 2):", boolean_array)
print("Sliced array containing elements > 2:", sliced_array)


# 48
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
condition = arr > 2
selected_elements = arr[condition]
print("Elements greater than 2 are :", selected_elements)


# 49
import numpy as np
arr_3d = np.array([[[1, 2, 3],
                    [4, 5, 6]],
                   [[7, 8, 9],
                    [10, 11, 12]],
                   [[13, 14, 15],
                    [16, 17, 18]]])
print("arr_3d\n",arr_3d)
sliced_arr = arr_3d[1:, :, :2]
print("\nsliced\n",sliced_arr)



# 50
import numpy as np
arr_2d = np.array([[1, 2, 3],
                   [4, 5, 6],
                   [7, 8, 9]])
condition = np.sum(arr_2d, axis=1) > 10

selected_rows = arr_2d[condition]
print(selected_rows)
```