1. [**2D Array Creation]**
   Write a NumPy program to create a 3x3 matrix with values ranging from 2 to 10.
   Expected Output:
   [[2 3 4]
   [5 6 7]
   [8 9 10]]

2. [**1D Array Creation**]
   Write a NumPy program to create an array with values ranging from 12 to 38.
   Expected Output:
   [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]

3. [**Reversing an Array**]
   Write a NumPy program to reverse an array (first element becomes last). Original array:
   [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]
   Reverse array:
   [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]

4. [**Make Border 1**]
   Write a NumPy program to create a 2d array with 1 on the border and 0 inside. Go to the editor
   Expected Output:
   Original array:
   [[1. 1. 1. 1. 1.]
   .....
   [1. 1. 1. 1. 1.]]
   1 on the border and 0 inside in the array
   [[1. 1. 1. 1. 1.]
   [1. 0. 0. 0. 1.]
   ……..
   [1. 0. 0. 0. 1.]
   [1. 1. 1. 1. 1.]]

5. [**Add border zero]**
   Write a NumPy program to add a border (filled with 0's) around an existing array.
   Expected Output:
   Original array:
   [[1. 1. 1.]
   [1. 1. 1.]
   [1. 1. 1.]]
   0 on the border and 1 inside in the array
   [[0. 0. 0. 0. 0.]

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

6. [**ChekerBoard Pattern**]
   Write a NumPy program to create a 8x8 matrix and fill it with a checkerboard pattern.
   Checkerboard pattern:
   [[0 1 0 1 0 1 0 1]
   .....
   [0 1 0 1 0 1 0 1]
   [1 0 1 0 1 0 1 0]]

7. [**Centigrade2Fahrenheit**]
   Write a NumPy program to convert the values of Centigrade degrees into Fahrenheit degrees. Centigrade values are stored into a NumPy array.
   Sample Array [0, 12, 45.21 ,34, 99.91]
   Expected Output:
   Values in Fahrenheit degrees:
   [0. 12. 45.21 34. 99.91]
   Values in Centigrade degrees:
   [-17.77777778 -11.11111111 7.33888889 1.11111111 37.72777778]

8. [**Membership Test**]
   Write a NumPy program to test whether each element of a 1-D array is also present in a second array.
   Expected Output:
   Array1: [0 10 20 40 60]
   Array2: [0, 40]
   Compare each element of array1 and array2
   [True False False True False]

9. [**Set Difference**]
   Write a NumPy program to find the set difference of two arrays. The set difference will return the sorted, unique values in array1 that are not in array2.
   Expected Output:
   Array1: [0 10 20 40 60 80]
   Array2: [10, 30, 40, 50, 70, 90]
   Set difference between two arrays:
   [0 20 60 80]

10. [**SymmetricDifference**]
    Write a NumPy program to find the set exclusive-or of two arrays. Set exclusive-or will return the sorted, unique values that are in only one (not both) of the input arrays.
    Array1: [0 10 20 40 60 80]
    Array2: [10, 30, 40, 50, 70]
    Unique values that are in only one (not both) of the input arrays:
    [0 20 30 50 60 70 80]

11. [**Array Comparison Operators**]
    Write a NumPy program to compare two given arrays.
    Array a: [1 2]
    Array b: [4 5]
    a > b
    [False False]
    a >= b
    [False False]
    a < b
    [True True]
    a <= b
    [True True]

12. [**Array2TxtFile**]
    Write a NumPy program to save a NumPy array to a text file.

13. [**toMultidimension**]
    Write a NumPy program to change the dimension of an array.
    Expected Output:
    9 rows and 0 columns
    (9,)
    (3, 3) -> 3 rows and 3 columns
    [[1 2 3]
    [4 5 6]
    [7 8 9]]
    Change array shape to (3, 3) -> 3 rows and 3 columns
    [[1 2 3]
    [4 5 6]
    [7 8 9]]

14. [**Reshaper**]
    Write a NumPy program to create a new shape to an array without changing its data.
    Reshape 3x2:

[[1 2]
[3 4]
[5 6]]
Reshape 2x3:
[[1 2 3]
[4 5 6]]

15. [**Formation of Diagonal Matrix**]
    Write a NumPy program to create a 2-D array whose diagonal equals [4, 5, 6, 8] and 0's elsewhere.
    Expected Output:
    [[4 0 0 0]
    [0 5 0 0]
    [0 0 6 0]
    [0 0 0 8]]

16. [**LinearLogSpace Data Generation**]
    Write a NumPy program to create a 1-D array of 20 element spaced evenly on a log scale between 2. and 5., exclusive.
    Expected Output:
    [100. 141.25375446 199.5262315 281.83829313
    .....
    25118.8643151 35481.33892336 50118.72336273 70794.57843841]

17. [**Frequency Distribution**]
    Write a NumPy program to count the frequency of unique values in NumPy array.
    Expected Output:
    Original array:
    [10 10 20 10 20 20 20 30 30 50 40 40]
    Frequency of unique values of the said array:
    [[10 20 30 40 50]
    [3 4 2 2 1]]

18. [**Array2CSV File**]
    Write a NumPy program to convert a NumPy array into a CSV file.

19. [**CSV File to Array**]
    Write a NumPy program to read a CSV data file and store records in an array.
    Sample CSV file: fdata.csv
    Date,Open,High,Low,Close
    03-10-16,774.25,776.065002,769.5,772.559998

.....
07-10-16,779.659973,779.659973,770.75,775.080017
Sample Output:
[(b'Date', nan, nan, nan, nan)
(b'03-10-16', 774.25, 776.065, 769.5 , 772.56)
.....
(b'07-10-16', 779.66, 779.66 , 770.75, 775.08)]

20. [**PILLOW2Array**]
    Write a NumPy program to convert a PILLOW Image into a NumPy array.
    Sample Output:
    [[[255 255 255 0]
    .....
    [255 255 255 0]]]

21. [**Array2Image**]
    Write a NumPy program to convert a NumPy array to an image. Display the image.
    Sample Output:
    test image

22. [**Clear NAN Values**]
    Write a NumPy program to remove nan values from a given array.
    Sample Output:
    Original array:
    [200. 300. nan nan nan 700.]
    After removing nan values:
    [200. 300. 700.]
    Original array:
    [[1. 2. 3.]
    [nan 0. nan]
    [6. 7. nan]]
    After removing nan values:
    [1. 2. 3. 0. 6. 7.]

23. [**CartesianProduct**]
    Write a NumPy program to create a Cartesian product of two arrays into single array of 2D points.
    Sample Output:
    [[1 4]
    .....
    [3 5]]

24. [**Histogram**]
    Write a NumPy program to compute the histogram of a set of data.
    Sample Output:
    Histogram image

25. [**Transpose**]
    Write a NumPy program to rearrange the dimensions of a given array.
    Sample Output:
    Original arrays:
    [[0 1 2 3]
    .....
    [20 21 22 23]]
    After reverse the dimensions:
    [[0 4 8 12 16 20]
    .....
    [3 7 11 15 19 23]]

26. [**Swap Columns**]
    Write a NumPy program to swap columns in a given array.
    Sample Output:
    Original array:
    [[0 1 2 3]
    [4 5 6 7]
    [8 9 10 11]]
    After swapping arrays:
    [[1 0 2 3]
    [5 4 6 7]
    [9 8 10 11]]

27. [**UpperTriangle**]
    Write a NumPy program to extract the upper triangular part of a NumPy matrix.
    Sample Output:
    Original array:
    [[0 1 2]
    .....
    [15 16 17]]
    Extract upper triangular part of the said array:
    [0 1 2 4 5 8]
    Extract upper triangular part of the said array:
    [0 1 4]

28. [**Averages of Triplets**]
    Write a NumPy program to create a new array which is the average of every consecutive triplet of elements of a given array.
    Sample Output:
    Original array:
    [1 2 3 2 4 6 1 2 12 0 -12 6]
    Average of every consecutive triplet of elements of the said array:
    [2. 4. 5. -2.]

29. [**k-Smallest Elements**]
    Write a NumPy program to find the k smallest values of a given NumPy array.
    Sample Output:
    Original arrays:
    [1. 7. 8. 2. 0.1 3. 15. 2.5]
    k smallest values:
    [0.1 1. 2. 2.5]