```
import numpy as np
```

cumsum()

Syntax: numpy.cumsum(array, axis=None, dtype=None, out=None)

where,

- **array:** The input array containing numbers whose cumulative sum is desired.
- **axis:** (Optional) The axis along which the cumulative sum is computed. If not specified, the array is flattened.
- **dtype:** (Optional) The data type of the returned array.
- **out:** (Optional) An alternative output array to place the result

1.Cumulative Sum of a One-Dimensional Array

```
array = np.array([1, 2, 3, 4, 5])
cumulative_sum = np.cumsum(array)

print("Original array:", array)
print("Cumulative sum:", cumulative_sum)
```

```
Original array: [1 2 3 4 5]
Cumulative sum: [ 1  3  6 10 15]
```

2.Cumulative Sum of a Two-Dimensional Array

```
# axis=0 refers to the rows (the vertical index).
# axis=1 refers to the columns (the horizontal index).

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

cumulative_sum_flattened = np.cumsum(array2)
cumulative_sum_axis0 = np.cumsum(array2, axis=0)
cumulative_sum_axis1 = np.cumsum(array2, axis=1)

print(cumulative_sum_flattened)
print()
print(cumulative_sum_axis0)
print()
print(cumulative_sum_axis1)
```

```
[ 1  3  6 10]
```

```
[[1 2]
 [4 6]]

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

```
# axis=0 refers to the rows (the vertical index).
# axis=1 refers to the columns (the horizontal index).

array2 = np.array([[1, 2],
                   [3, 4],
                   [5, 6],
                   [7, 8]])

cumulative_sum_flattened = np.cumsum(array2)
cumulative_sum_axis0 = np.cumsum(array2, axis=0)
cumulative_sum_axis1 = np.cumsum(array2, axis=1)

print(cumulative_sum_flattened)
print()
print(cumulative_sum_axis0)
print()
print(cumulative_sum_axis1)
```

```
[ 1  3  6 10 15 21 28 36]

[[ 1  2]
 [ 4  6]
 [ 9 12]
 [16 20]]

[[ 1  3]
 [ 3  7]
 [ 5 11]
 [ 7 15]]
```

3.Cumulative Sum with a Specified Data Type

```
array3 = np.array([1, 2, 3])
cumulative_sum_float = np.cumsum(array3, dtype=float)
print(cumulative_sum_float)

# Output: [1.0 3.0 6.0]
```

```
[1. 3. 6.]
```

4.Handling Missing Values (NaN) with `cumsum()`

```
array_with_nan_cumsum = np.array([1, 2, np.nan, 4, 5])
cumulative_sum_with_nan = np.cumsum(array_with_nan_cumsum)

```

```
print("Original array with NaN:", array_with_nan_cumsum)
print("Cumulative sum with NaN:", cumulative_sum_with_nan)
```

```
Original array with NaN: [ 1.  2. nan  4.  5.]
Cumulative sum with NaN: [ 1.  3. nan nan nan]
```

```
array_2d_with_nan_cumsum = np.array([[1, 2, np.nan],
                                     [4, np.nan, 6],
                                     [np.nan, 8, 9]])

print("Original 2D array with NaN:\n", array_2d_with_nan_cumsum)

cumulative_sum_2d_axis0 = np.cumsum(array_2d_with_nan_cumsum, axis=0)
print("\nCumulative sum (axis=0):\n", cumulative_sum_2d_axis0)

cumulative_sum_2d_axis1 = np.cumsum(array_2d_with_nan_cumsum, axis=1)
print("\nCumulative sum (axis=1):\n", cumulative_sum_2d_axis1)
```

```
Original 2D array with NaN:
 [[ 1.  2. nan]
 [ 4. nan  6.]
 [nan  8.  9.]]

Cumulative sum (axis=0):
 [[ 1.  2. nan]
 [ 5. nan nan]
 [nan nan nan]]

Cumulative sum (axis=1):
 [[ 1.  3. nan]
 [ 4. nan nan]
 [nan nan nan]]
```

cumprod()

1.Cumulative Product of a One-Dimensional Array

```
array = np.array([1, 2, 3, 4, 5])
cumulative_product = np.cumprod(array)

print("Original array:", array)
print("Cumulative product:", cumulative_product)
```

```
Original array: [1 2 3 4 5]
Cumulative product: [  1   2   6  24 120]
```

2.Cumulative Product of a Two-Dimensional Array

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

cumulative_product_flattened = np.cumprod(array2)
cumulative_product_axis0 = np.cumprod(array2, axis=0)
cumulative_product_axis1 = np.cumprod(array2, axis=1)

print("Cumulative product (flattened):\n", cumulative_product_flattened)
print()
print("Cumulative product (axis=0):\n", cumulative_product_axis0)
print()
print("Cumulative product (axis=1):\n", cumulative_product_axis1)
```

```
Cumulative product (flattened):
 [ 1  2  6 24]

Cumulative product (axis=0):
 [[1 2]
 [3 8]]

Cumulative product (axis=1):
 [[ 1  2]
 [ 3 12]]
```

3.Cumulative Product with a Specified Data Type

```
array3 = np.array([1, 2, 3])
cumulative_product_float = np.cumprod(array3, dtype=float)
print("Cumulative product (dtype=float):", cumulative_product_float)
```

```
Cumulative product (dtype=float): [1. 2. 6.]
```

5.Handling Missing Values (NaN) with `cumprod()`

```
array_with_nan_cumprod = np.array([1, 2, np.nan, 4, 5])
cumulative_product_with_nan = np.cumprod(array_with_nan_cumprod)

print("Original array with NaN:", array_with_nan_cumprod)
print("Cumulative product with NaN:", cumulative_product_with_nan)
```

```
Original array with NaN: [ 1.  2. nan  4.  5.]
Cumulative product with NaN: [ 1.  2. nan nan nan]
```

```
array_2d_with_nan_cumprod = np.array([[1, 2, np.nan],
                                      [4, np.nan, 6],
                                      [np.nan, 8, 9]])

print("Original 2D array with NaN:\n", array_2d_with_nan_cumprod)

cumulative_product_2d_axis0 = np.cumprod(array_2d_with_nan_cumprod, axis=0)
print("\nCumulative product (axis=0):\n", cumulative_product_2d_axis0)
```

```
9
10 cumulative_product_2d_axis1 = np.cumprod(array_2d_with_nan_cumprod, axis=1)
11 print("\nCumulative product (axis=1):\n", cumulative_product_2d_axis1)
```

```
Original 2D array with NaN:
 [[ 1.  2. nan]
 [ 4. nan  6.]
 [nan  8.  9.]]

Cumulative product (axis=0):
 [[ 1.  2. nan]
 [ 4. nan nan]
 [nan nan nan]]

Cumulative product (axis=1):
 [[ 1.  2. nan]
 [ 4. nan nan]
 [nan nan nan]]
```

Cumulative Sum and Product with Pandas DataFrames

cumsum() and cumprod() methods for data frame ignore missing values but preserve them in the resulting arrays.

- When summing the data, missing values will be treated as zero
- If all values are missing, the sum will be equal to NaN

```
1 import pandas as pd
```

```
1 df = pd.DataFrame({
2     'col1': [1, 2, np.nan, 4, 5],
3     'col2': [10, np.nan, 30, 40, 50]
4 })
5
6 print("Original DataFrame:\n", df)
7
8 cumulative_sum_df = df.cumsum()
9 print("\nCumulative sum of DataFrame:\n", cumulative_sum_df)
10
11 cumulative_prod_df = df.cumprod()
12 print("\nCumulative product of DataFrame:\n", cumulative_prod_df)
```

```
Original DataFrame:
    col1  col2
0   1.0  10.0
1   2.0   NaN
2   NaN  30.0
3   4.0  40.0
4   5.0  50.0

Cumulative sum of DataFrame:
    col1   col2
```

```
0   1.0   10.0
1   3.0    NaN
2   NaN   40.0
3   7.0   80.0
4  12.0  130.0

Cumulative product of DataFrame:
    col1      col2
0   1.0      10.0
1   2.0       NaN
2   NaN     300.0
3   8.0   12000.0
4  40.0  600000.0
```