

Prev and Next permutations:

Prev Permutation: Lexicographically largest permutation of the given array which is less than the given array.

Next Permutation: Lexicographically smallest permutation of the array which is larger than the given array.

prev_permutations function changes the array to its prev permutation and returns 0 if the final array is sorted.

Prev and Next Permutations

Array :

Prev PermutationNext PermutationClearBack

[Copy Text](#)

Prev permutation of the array is:

3 1 4 2

Other prev permutations are:

3 1 2 4

2 4 3 1

2 4 1 3

2 3 4 1

2 3 1 4

2 1 4 3

2 1 3 4

1 4 3 2

1 4 2 3

1 3 4 2

1 3 2 4

1 2 4 3

1 2 3 4

The prev permutation generates the array until it becomes sorted.

Prev and Next Permutations

Array :

[Copy Text](#)

```
Next permutation of the array is:  
3 2 4 1  
Other next permutations are:  
3 4 1 2  
3 4 2 1  
4 1 2 3  
4 1 3 2  
4 2 1 3  
4 2 3 1  
4 3 1 2  
4 3 2 1
```

Next permutation permutes the array until it becomes reverse sorted.

Using it in calculator:

Initialize the array

Call the function

Write the name of arr

[Explore the docs »](#)

Calculator

Enter the Expression below

```
arr=[2,1];prev_permutation(arr);arr
```

[Copy Text](#)

```
1,2
```

The previous permutation of [2,1] is [1,2] because among all permutations of [2,1] [1,2] is largest and is smaller than [2,1]

[Explore the docs »](#)

Calculator

Enter the Expression below

```
arr=[4,3,2,1];prev_permutation(arr);arr
```

Evaluate

Reset

Back

Copy Text

```
4,3,1,2
```

The previous permutation of [4,3,2,1] is 4 3 1 2

[Explore the docs »](#)

Calculator

Enter the Expression below

```
arr=[1,2,3,4];next_permutation(arr);arr
```

Evaluate

Reset

Back

Copy Text

```
1,2,4,3
```