

The Array class is the base class for all the arrays in C#. It is defined in the System namespace. The Array class provides various properties and methods to work with arrays.

## Properties of the Array Class

The following table describes some of the most commonly used properties of the Array class:

| Sr.No | Property  |
|-------|---|
| 1     | <b>IsFixedSize</b><br>Gets a value indicating whether the Array has a fixed size.   |
| 2     | <b>IsReadOnly</b><br>Gets a value indicating whether the Array is read-only.  |
| 3     | <b>Length</b><br>Gets a 32-bit integer that represents the total number of elements in all the dimensions of the Array.     |
| 4     | <b>LongLength</b><br>Gets a 64-bit integer that represents the total number of elements in all the dimensions of the Array. |
| 5     | <b>Rank</b><br>Gets the rank <i>numberofdimensions</i> of the Array.  |

## Methods of the Array Class

The following table describes some of the most commonly used methods of the Array class:

| Sr.No | Methods   |
|-------|---|
| 1     | <b>Clear</b><br>Sets a range of elements in the Array to zero, to false, or to null, depending on the element type.   |
| 2     | <b>CopyArray, Array, Int32</b><br>Copies a range of elements from an Array starting at the first element and pastes them into another Array starting at the first element. The length is specified as a 32-bit integer. |
| 3     | <b>CopyToArray, Int32</b><br>Copies all the elements of the current one-dimensional Array to the specified one-   |

dimensional Array starting at the specified destination Array index. The index is specified as a 32-bit integer.

4      **GetLength**

Gets a 32-bit integer that represents the number of elements in the specified dimension of the Array.

5      **GetLongLength**

Gets a 64-bit integer that represents the number of elements in the specified dimension of the Array.

6      **GetLowerBound**

Gets the lower bound of the specified dimension in the Array.

7      **GetType**

Gets the Type of the current instance. *Inherited from Object.*

8      **GetUpperBound**

Gets the upper bound of the specified dimension in the Array.

9      **GetValueInt32**

Gets the value at the specified position in the one-dimensional Array. The index is specified as a 32-bit integer.

10     **IndexOfArray, Object**

Searches for the specified object and returns the index of the first occurrence within the entire one-dimensional Array.

11     **ReverseArray**

Reverses the sequence of the elements in the entire one-dimensional Array.

12     **SetValueObject, Int32**

Sets a value to the element at the specified position in the one-dimensional Array. The index is specified as a 32-bit integer.

13     **SortArray**

Sorts the elements in an entire one-dimensional Array using the IComparable implementation of each element of the Array.

14     **ToStringk**

Returns a string that represents the current object. *Inherited from Object.*

For complete list of Array class properties and methods, please consult Microsoft documentation on C#.

## Example

The following program demonstrates use of some of the methods of the Array class:

```
using System;
namespace ArrayApplication
{
    class MyArray
    {
        static void Main(string[] args)
        {
            int[] list = { 34, 72, 13, 44, 25, 30, 10 };
            int[] temp = list;
            Console.WriteLine("Original Array: ");

            foreach (int i in list)
            {
                Console.Write(i + " ");
            }
            Console.WriteLine();

            // reverse the array
            Array.Reverse(temp);
            Console.WriteLine("Reversed Array: ");

            foreach (int i in temp)
            {
                Console.Write(i + " ");
            }
            Console.WriteLine();

            //sort the array
            Array.Sort(list);
            Console.WriteLine("Sorted Array: ");

            foreach (int i in list)
            {
                Console.Write(i + " ");
            }
            Console.WriteLine();
            Console.ReadKey();
        }
    }
}
```

When the above code is compiled and executed, it produces the following result:

```
Original Array: 34 72 13 44 25 30 10
Reversed Array: 10 30 25 44 13 72 34
Sorted Array: 10 13 25 30 34 44 72
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js