**C# Jagged Array**

In this tutorial, we will learn about the C# jagged array. We will learn to declare, initialize, and access the jagged array with the help of examples.

In C#, a jagged array consists of multiple arrays as its element. However, unlike multidimensional arrays, each array inside a jagged array can be of different sizes.

Before you learn about jagged array, make sure to know about

* [C# Arrays](https://www.programiz.com/csharp-programming/arrays)
* [C# Multidimensional Arrays](https://www.programiz.com/csharp-programming/multidimensional-arrays)

**C# Jagged Array Declaration**

Here's a syntax to declare a jagged array in C#.

dataType[ ][ ] nameOfArray = new dataType[rows][ ];

Let's see an example,

// declare jagged array

int[ ][ ] jaggedArray = new int[2][ ];

Here,

* int - data type of the array
* [][] - represents jagged array
* jaggedArray - name of the jagged array
* [2][] - represents the number of elements (arrays) inside the jagged array

Since we know each element of a jagged array is also an array, we can set the size of the individual array. For example,

// set size of the first array as 3

jaggedArray[0] = new int[3];

// set size of second array as 2

jaggedArray[1] = new int[2];

**Initializing Jagged Array**

There are different ways to initialize a jagged array. For example,

**1. Using the index number**

Once we declare a jagged array, we can use the index number to initialize it. For example,

// initialize the first array

jaggedArray[0][0] = 1;

jaggedArray[0][1] = 3;

jaggedArray[0][2] = 5;

// initialize the second array

jaggedArray[1][0] = 2;

jaggedArray[1][1] = 4;

Here,

* index at the first square bracket represents the index of the jagged array element
* index at the second square bracket represents the index of the element inside each element of the jagged array

**2. Initialize without setting size of array elements**

// declaring string jagged array

int[ ][ ] jaggedArray = new int[2] [ ];

// initialize each array

jaggedArray[0] = new int[] {1, 3, 5};

jaggedArray[1] = new int[] {2, 4};

**3. Initialize while declaring Jagged Array**

int[ ][ ] jaggedArray = {

new int[ ] {10, 20, 30},

new int[ ] {11, 22},

new int[ ] {88, 99}

};

**Accessing elements of a jagged array**

We can access the elements of the jagged array using the index number. For example,

// access first element of second array

jaggedArray[1][0];

// access second element of the second array

jaggedArray[1][1];

// access second element of the first array

jaggedArray[0][1];

**Example: C# Jagged Array**

using System;

namespace JaggedArray {

class Program {

static void Main(string[] args) {

// create a jagged array

int[ ][ ] jaggedArray = {

new int[] {1, 3, 5},

new int[] {2, 4},

};

// print elements of jagged array

Console.WriteLine("jaggedArray[1][0]: " + jaggedArray[1][0]);

Console.WriteLine("jaggedArray[1][1]: " + jaggedArray[1][1]);

Console.WriteLine("jaggedArray[0][2]: " + jaggedArray[0][2]);

Console.ReadLine();

}

}

}

**Output**

jaggedArray[1][0]: 2

jaggedArray[1][1]: 4

jaggedArray[0][2]: 5

Here, inside a jagged array,

* jaggedArray[1][0] - first element of the second array
* jaggedArray[1][1] - second element of the second array
* jaggedArray[0][2] - third element of the first array

**Iterating through a jagged array**

In C#, we can use loops to iterate through each element of the jagged array. For example,

using System;

namespace JaggedArray {

class Program {

static void Main(string[] args) {

// declare a jagged array

int[][] jaggedArray = new int[2][];

// set size of Jagged Array Elements

jaggedArray[0] = new int[3];

jaggedArray[1] = new int[2];

// initialize the first array

jaggedArray[0][0] = 1;

jaggedArray[0][1] = 3;

jaggedArray[0][2] = 5;

// initialize the second array

jaggedArray[1][0] = 2;

jaggedArray[1][1] = 2;

// outer for loop

for (int i = 0; i < jaggedArray.Length; i++) {

Console.Write("Element "+ i +": ");

// inner for loop

for (int j = 0; j < jaggedArray[i].Length; j++) {

Console.Write(jaggedArray[i][j] + " ");

}

Console.WriteLine();

}

Console.ReadLine();

}

}

}

**Output**

Element 0: 1 3 5

Element 1: 2 2

In the above example, we have used a [nested for loop](https://www.programiz.com/csharp-programming/nested-loops) to iterate through the jagged array. Here,

**1. Outer for loop**

* to access the elements (arrays) of the jagged array
* jaggedArray.Length - gives the size of jagged array

**2. Inner for loop**

* to access the elements of the individual array inside the jagged array.
* jaggedArray[i].Length - gives the size of elements of the ith array inside the jagged array

**Jagged Array with Multidimensional Array**

In C#, we can also use multidimensional arrays as Jagged Array Elements. For example,

int[ ][ , ] jaggedArrayTwoD = new int[2][ , ] {

new int[,] { {1, 8}, {6, 7} },

new int[,] { {0, 3}, {5, 6}, {9, 10} }

};

Here, each element of the jagged array is a multidimensional array:

* new int[,] { {1, 8}, {6, 7} } - 2D array with 2 elements
* new int[ , ] { {0, 3}, {5, 6}, {9, 10} } - 2D array with 3 elements

Let's see an example,

using System;

namespace JaggedArray {

class Program {

static void Main(string[] args) {

// declare and initialize jagged array with 2D array

int[][,] jaggedArray = new int[3][ , ] {

new int[ , ] { {1, 8}, {6, 7} },

new int[ , ] { {0, 3}, {5, 6}, {9, 10} },

new int[ , ] { {11, 23}, {100, 88}, {0, 10} }

};

Console.WriteLine(jaggedArray[0][0, 1]);

Console.WriteLine(jaggedArray[1][2, 1]);

Console.WriteLine(jaggedArray[2][1, 0]);

Console.ReadLine();

}

}

}

**Output**

8

10

100

In the above example, notice the code,

jaggedArray[0][0, 1]

Here,

* [0] - represents the first element (2D array) of the jagged array
* [0, 1] - represents the second element of the first array inside the 2D array