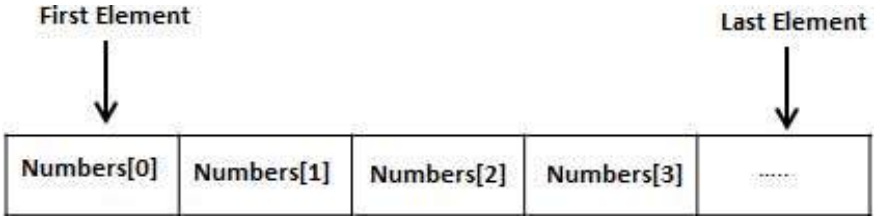
Array (Array) in C

An array stores a set of fixed-size elements in the same type. An array is used to store a data set, but it is often more useful to think of an array as a set of variables of the same type stored in adjacent memory locations.

In C #, arrays store a set of fixed-size elements in the same type. An array is used to store a data set, but it is often more useful to think of an array as a set of variables of the same type stored in adjacent memory locations.

Instead of declaring variables in a discrete way, like variables number0, number1 . and number99, you can declare an array of values ??like numbers [0], numbers [1] and . numbers [99] to represent prices treatment separately. A specific member of the array can be accessed via index (index).

All arrays include adjacent memory locations. The lowest address corresponds to the first member and the highest address corresponding to the last member of the array.



Declaring an array in C #

To declare an array in the C # language, you can use the syntax:

```
ki ? u_d ? _li ? u []  t ê n_m ? ng ;
```

In the above syntax:

1. *Data_type* is used to specify the type of element in the array.
2. [] rank determination or size of array.
3. *array_name* identifies the array name.

For example:

```
double []  balance ;
```

Initialize arrays in C #

Declaring an array does not initialize arrays in memory. When the array variable is initialized, you can assign values ??to that array.

Arrays are a reference type, so you need to use the **new** keyword in C # to create an Instance of that array. For example:

```
double []  balance  = new double [ 10 ];
```

Assign values ??to an array in C #

You can assign values ??to individual array elements using the array index, such as:

```
double []  balance  = new double [ 10 ];  balance [ 0 ] = 4500.0 ;
```

You can assign values ??to arrays at the time of array declaration, as follows:

```
double []  balance  = { 2340.0 , 4523.69 , 3421.0 };
```

You can also create and declare an array, as follows:

```
int []  marks  = new int [ 5 ] { 99 , 98 , 92 , 97 , 95 };
```

You can also skip array sizes, like:

```
int []  marks  = new int [] { 99 , 98 , 92 , 97 , 95 };
```

You can copy an array variable into another target array variable. In this situation, both the target variable and the source variable point to the same memory location:

```
int []  marks  = new int [] { 99 , 98 , 92 , 97 , 95 }; int []  score  =  marks
```

When you create an array, the default C # compiler initializes each array element to a default value depending on the array type. For example, for an int array, all elements are initialized to 0.

Access array elements in C #

An element is accessed by the array index. This is done by placing the index of the element inside the square brackets after the array name. For example:

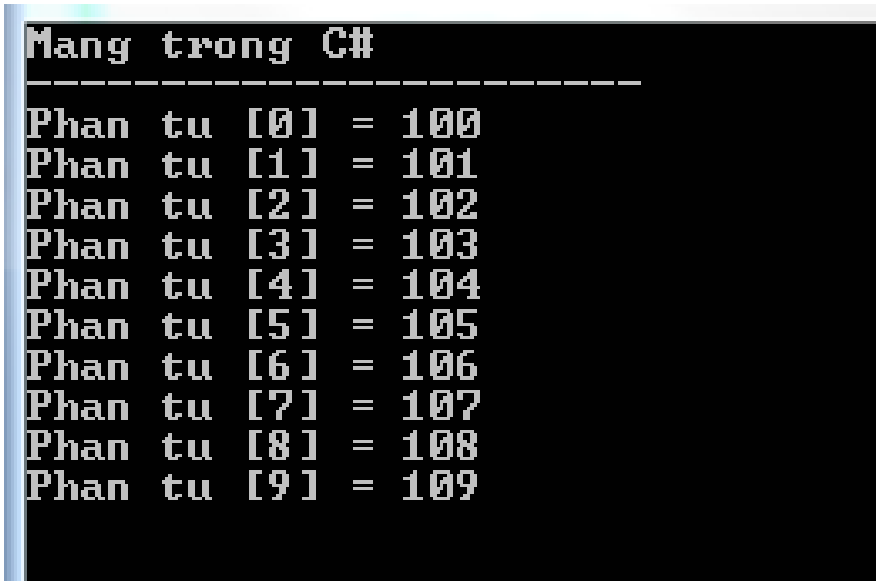
```
double  salary  =  balance [ 9 ];
```

The following example illustrates the concept of declaring, assigning and accessing arrays in C # mentioned above:

```
using System ; namespace QTMCsharp { class TestCsharp { static void Main ( stri
```

If you do not use the **Console.ReadKey ()** command; then the program will run and finish (so fast that you can not see the results). This command allows us to see the results more clearly.

Compiling and running the above C # program will produce the following results:



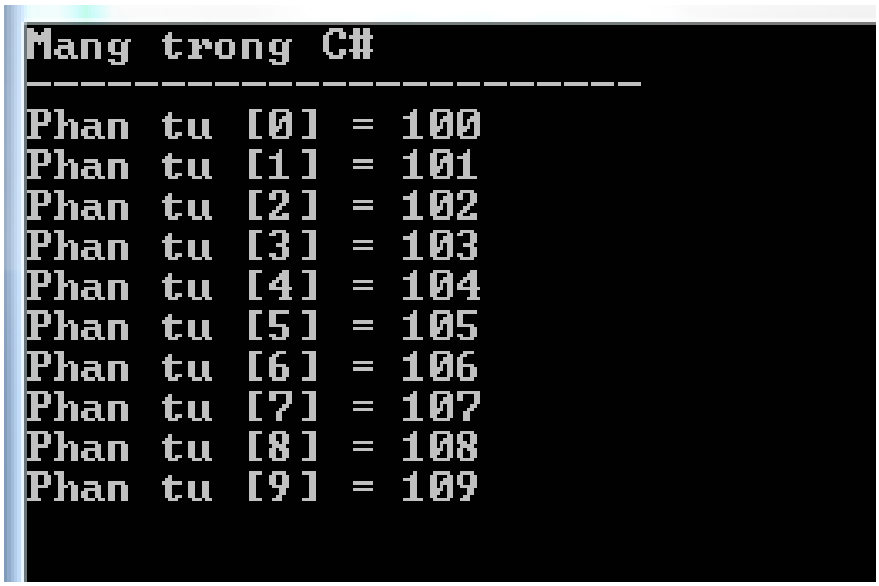
Use foreach loop in C #

In the previous example, we used a for loop to access each element in the array. You can also use a foreach command to browse an array in C #:

```
using System ; namespace QTMCsharp { class TestCsharp { static void Main ( stri
```

If you do not use the **Console.ReadKey ()** command; then the program will run and finish (so fast that you can not see the results). This command allows us to see the results more clearly.

Compiling and running the above C # program will produce the following results:



Array details in C #

Arrays are a very important part of the C # language. Here are the important array definitions that are presented more clearly for C # programmers:

Concept Description
Multidimensional array in C #

C # supports multidimensional arrays. The simplest pattern of multidimensional arrays is a two-dimensional array
Jagged array in C #

C # supports multidimensional arrays, which are arrays of arrays
Passing array to function in C #

You can pass a function pointer to an array by specifying the array name without the array index
Parameter array in C #

Used to pass an unknown number of parameters to a function
Array class in C #
Defined in the System namespace, it is the base class for all arrays, and provides properties and methods for working with arrays.

Follow tutorialspoint

Previous article: Converting data types in C #

Next lesson: String (String) in C #