**泛型和数组（C# 编程指南）**

在 C# 2.0 以及更高版本中，下限为零的一维数组自动实现 [IList<T>](https://msdn.microsoft.com/zh-cn/library/5y536ey6.aspx)。这使您可以创建能够使用相同代码循环访问数组和其他集合类型的泛型方法。此技术主要对读取集合中的数据很有用。 [IList<T>](https://msdn.microsoft.com/zh-cn/library/5y536ey6.aspx) 接口不能用于在数组中添加或移除元素。如果尝试对此上下文中的数组调用 [IList<T>](https://msdn.microsoft.com/zh-cn/library/5y536ey6.aspx) 方法（例如 [RemoveAt](https://msdn.microsoft.com/zh-cn/library/c93ab5c9.aspx)），则将引发异常。

下面的代码示例演示带有 [IList<T>](https://msdn.microsoft.com/zh-cn/library/5y536ey6.aspx) 输入参数的单个泛型方法如何同时循环访问列表和数组，本例中为整数数组。

class Program

{

static void Main()

{

int[] arr = { 0, 1, 2, 3, 4 };

List<int> list = new List<int>();

for (int x = 5; x < 10; x++)

{

list.Add(x);

}

ProcessItems<int>(arr);

ProcessItems<int>(list);

}

static void ProcessItems<T>(IList<T> coll)

{

// IsReadOnly returns True for the array and False for the List.

System.Console.WriteLine

("IsReadOnly returns {0} for this collection.",

coll.IsReadOnly);

// The following statement causes a run-time exception for the

// array, but not for the List.

//coll.RemoveAt(4);

foreach (T item in coll)

{

System.Console.Write(item.ToString() + " ");

}

System.Console.WriteLine();

}

}

**请参阅**

[System.Collections.Generic](https://msdn.microsoft.com/zh-cn/library/system.collections.generic.aspx)

[C# 编程指南](https://msdn.microsoft.com/zh-cn/library/67ef8sbd.aspx)

[泛型（C# 编程指南）](https://msdn.microsoft.com/zh-cn/library/512aeb7t.aspx)

[数组（C# 编程指南）](https://msdn.microsoft.com/zh-cn/library/9b9dty7d.aspx)

[.NET Framework 中的泛型](https://msdn.microsoft.com/zh-cn/library/ms172192.aspx)