C#中的数组为对象.

例:程序array\_test1

// Copyright 2016.刘珅珅

// author：刘珅珅

// 数组测试

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace array\_test1

{

class ArrayTest

{

static void Main(string[] args)

{

int[] nums = {99, 10, 100, 18, 78, 23, 63, 9, 87, 49 };

int[] nums1 = new int[] { 99, 10, 100, 18, 78, 23, 63, 9, 87, 49 };

int[] nums2 = new int[10];

nums2[0] = 99;

nums2[1] = 10;

nums2[2] = 100;

nums2[3] = 18;

nums2[4] = 78;

nums2[5] = 23;

nums2[6] = 63;

nums2[7] = 9;

nums2[8] = 87;

nums2[9] = 49;

// 多维数组

// 创建一个行为3，列为4的二维数组

int[,] table = new int[3, 4];

Console.WriteLine("table length is {0}", table.Length); // 12

Console.WriteLine("table dimension count is {0}", table.Rank); // 2

Console.WriteLine("table row count is {0}", table.GetLength(0));

Console.WriteLine("table column count is {0}", table.GetLength(1));

// 遍历

for (int t = 0; t < table.GetLength(0); ++t)

{

for (int i = 0; i < table.GetLength(1); ++i)

{

table[t, i] = t \* 4 + i + 1;

Console.Write(table[t, i] + " ");

}

Console.WriteLine();

}

// 2X2数组

int[,] table1 = new int[,] {{1, 2}, {3, 4}};

// 遍历

for (int t = 0; t < table1.GetLength(0); ++t)

{

for (int i = 0; i < table1.GetLength(1); ++i )

{

Console.Write(table1[t, i] + " ");

}

Console.WriteLine();

}

// 交错数组

// 不规则的数组，当成一维数组

int[][] table2 = new int[3][];

table2[0] = new int[] { 1, 2, 3, 4, 5};

table2[1] = new int[] { 6, 7, 8 };

table2[2] = new int[] { 10, 11, 12, 13 };

Console.WriteLine("table2 length is {0}", table2.Length); // 3

Console.WriteLine("table2 dimension count is {0}", table2.Rank); // 1

Console.WriteLine("table2 row count is {0}", table2.GetLength(0)); // 3

// Console.WriteLine("column count is {0}", table2.GetLength(1)); // 异常

// 遍历

for (int i = 0; i < table2.GetLength(0); ++i)

{

for (int j = 0; j < table2[i].GetLength(0); ++j)

{

Console.Write(table2[i][j] + " ");

}

Console.WriteLine();

}

// 交错数组

// 更不规则的数组，仍然被看成一维数组

int[][,] table3 = new int[3][,]

{

new int[,] {{1, 3}, {5, 7}},

new int[,] {{0, 2}, {4, 6}, {8, 10}},

new int[,] {{11, 22, 33}, {44, 55, 66}, {77, 88, 99}}

};

Console.WriteLine("table3 length is {0}", table3.Length); // 3

Console.WriteLine("table3 dimension count is {0}", table3.Rank); // 1

Console.WriteLine("table3 row count is {0}", table3.GetLength(0)); // 3

// 遍历

for (int i = 0; i < table3.GetLength(0); ++i)

{

// Console.WriteLine("table3[{0}] dimension count is {1}", i, table3[i].Rank); // 2

for (int j = 0; j < table3[i].GetLength(0); ++j)

{

for (int k = 0; k < table3[i].GetLength(1); ++k)

{

Console.Write(table3[i][j, k] + " ");

}

}

Console.WriteLine();

}

int[, ,] many\_dim = new int[3, 4, 5];

Console.WriteLine("many\_dim dimension count is {0}", many\_dim.Rank);

}

}

}

C#中允许将数组直接赋给数组，C++不允许这样。

例：程序array\_test2

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace array\_test2

{

class ArrayTest

{

static void Main(string[] args)

{

int[] nums1 = new int[10];

int[] nums2 = new int[] { -1, -2, -3, -4, - 5, -6, -7, -8, -9, -10};

for (int i = 0; i < 10; ++i)

{

nums1[i] = i + 1;

}

Console.Write("nums1 is : ");

// 1 2 3 4 5 6 7 8 9 10

for (int i = 0; i < 10; ++i)

{

Console.Write(nums1[i] + " ");

}

Console.WriteLine();

Console.Write("nums2 is : ");

// -1 -2 -3 -4 -5 -6 -7 -8 -9 -10

for (int i = 0; i < 10; ++i)

{

Console.Write(nums2[i] + " ");

}

Console.WriteLine();

nums2 = nums1; // C#中允许这样赋值

Console.Write("nums1 is : ");

// 1 2 3 4 5 6 7 8 9 10

for (int i = 0; i < 10; ++i)

{

Console.Write(nums1[i] + " ");

}

Console.WriteLine();

Console.Write("nums2 is : ");

// 1 2 3 4 5 6 7 8 9 10

for (int i = 0; i < 10; ++i)

{

Console.Write(nums2[i] + " ");

}

Console.WriteLine();

nums2[3] = 99;

Console.Write("nums1 is : ");

// 1 2 3 99 5 6 7 8 9 10

for (int i = 0; i < 10; ++i)

{

Console.Write(nums1[i] + " ");

}

Console.WriteLine();

Console.Write("nums2 is : ");

// 1 2 3 99 5 6 7 8 9 10

for (int i = 0; i < 10; ++i)

{

Console.Write(nums2[i] + " ");

}

Console.WriteLine();

}

}

}

隐式类型的数组

var jag = new[] {

new[] {1, 2, 3, 4},

new[] {1, 3}

};

隐式类型声明的数组元素的类型必须相同。