**6 kyu**

**Sorted Arrays**

26692% of 8040 of323[dnolan](https://www.codewars.com/users/dnolan)

C#

* [TRAIN AGAIN](https://www.codewars.com/kata/sorted-arrays/train/csharp)
* [NEXT KATA](https://www.codewars.com/trainer/csharp)

Details

[Solutions](https://www.codewars.com/kata/sorted-arrays/solutions/csharp)

[Forks (1)](https://www.codewars.com/kata/sorted-arrays/forks/csharp)

[Discourse (22)](https://www.codewars.com/kata/sorted-arrays/discuss/csharp)

* Add to Collection
* |
* Share this kata:

Given any number of arrays each sorted in *ascending order*, find the nth smallest number of all their elements.

All the arguments except the last will be arrays, the last argument is n.

nthSmallest([1,5], [2], [4,8,9], 4) // returns 5 because it's the 4th smallest value

Be mindful of performance.

<https://www.codewars.com/kata/sorted-arrays/csharp>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp1

{

class Program

{

public static int NthSmallest(int[][] arr, int n)

{

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

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

{

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

{

lista.Add(arr[i][j]);

}

}

lista.Sort();

return lista[n];

}

static void Main(string[] args)

{

Console.ReadLine();

}

}

}

**using System.Collections.Generic;**

**public class Kata**

**{**

**public static int NthSmallest(int[][] arr, int n)**

**{**

**var sorted = new List<int>();**

**foreach (var a in arr)**

**{**

**sorted = Merge(sorted, a);**

**}**

**return sorted[n - 1];**

**}**

**public static List<int> Merge(List<int> sorted, int[] input)**

**{**

**var output = new List<int>();**

**int indexSorted = 0;**

**int indexInput = 0;**

**while (indexSorted < sorted.Count && indexInput < input.Length)**

**{**

**if (sorted[indexSorted] < input[indexInput])**

**{**

**output.Add(sorted[indexSorted]);**

**++indexSorted;**

**}**

**else**

**{**

**output.Add(input[indexInput]);**

**++indexInput;**

**}**

**}**

**while (indexSorted < sorted.Count)**

**{**

**output.Add(sorted[indexSorted]);**

**++indexSorted;**

**}**

**while (indexInput < input.Length)**

**{**

**output.Add(input[indexInput]);**

**++indexInput;**

**}**

**return output;**

**}**

**}**