**5 kyu**

**From..To..Series #2: from arr1 to arr2. Find the most same sum value of pairs**

4194% of 2758 of59[myjinxin2015](https://www.codewars.com/users/myjinxin2015)

JavaScript

* [TRAIN AGAIN](https://www.codewars.com/kata/from-dot-to-dot-series-number-2-from-arr1-to-arr2-find-the-most-same-sum-value-of-pairs/train/javascript)
* [NEXT KATA](https://www.codewars.com/trainer/javascript)

Details

[Solutions](https://www.codewars.com/kata/from-dot-to-dot-series-number-2-from-arr1-to-arr2-find-the-most-same-sum-value-of-pairs/solutions/javascript)

[Discourse (20)](https://www.codewars.com/kata/from-dot-to-dot-series-number-2-from-arr1-to-arr2-find-the-most-same-sum-value-of-pairs/discuss/javascript)

* Add to Collection
* |
* Share this kata:

**Description:**

Give you two array arr1 and arr2. They contains some numbers. They have the same length. Calculate the sum of the corresponding position number, find the most pairs that has the same sum value. return the result by a 2D array that contains all pairs. For example:

arr1=[1,2,3,4,5]

arr2=[9,8,0,0,0]

findPair(arr1,arr2) should return [[1,9],[2,8]]

because:

1 2 3 4 5

+ + + + +

9 8 0 0 0

| | | | |

10 10 3 4 5

1,9 and 2,8 have the same sum value 10

You do not need to verify the input data. arr1 and arr2 will always have the same number of elements, and testcase will not provide an empty array.

If all pairs has diffrent sum value, please return [].

If more than one groups of pairs has same number, choose which has the max sum value(see the last example)

You don't need sort the result, return the pairs according to the order of the index.

**Examples**

arr1=[1,2,3,4,5]

arr2=[0,0,0,0,0]

findPair(arr1,arr2) should return []

arr1=[1,2,3,4,5]

arr2=[5,4,3,2,1]

findPair(arr1,arr2) should return [[1,5],[2,4],[3,3],[4,2],[5,1]]

arr1=[0,1,3,4,5]

arr2=[1,0,3,2,1]

findPair(arr1,arr2) should return [[3,3],[4,2],[5,1]]

arr1=[1,2,3,4,5]

arr2=[-1,2,-3,4,-5]

findPair(arr1,arr2) should return [[1,-1],[3,-3],[5,-5]]

arr1=[1,2,3,0,5,-2]

arr2=[-1,2,-3,4,-5,6]

findPair(arr1,arr2) should return [[2,2],[0,4],[-2,6]]

<https://www.codewars.com/kata/from-dot-to-dot-series-number-2-from-arr1-to-arr2-find-the-most-same-sum-value-of-pairs/javascript>

1. **function** findPair(arr1,arr2){
2. *//coding and coding..*
3. dic = {};
4. frec\_sumas = {};
6. **var** sum = 0;
7. **var** max\_sum = 0;
8. **var** max\_frec = 0;
10. **for** (let i = 0; i < arr1.length; i++)
11. {
12. sum = arr1[i] + arr2[i];
13. max\_sum = Math.max(max\_sum, sum);
14. **if** (dic.hasOwnProperty(sum))
15. {
16. par = [];
17. par.push(arr1[i]);
18. par.push(arr2[i]);
19. dic[sum].push(par);
20. }
21. **else**
22. {
23. dic[sum] = [];
24. *//dic[sum].Add(new int[] { arr1[i], arr2[i] });*
25. par = [];
26. par.push(arr1[i]);
27. par.push(arr2[i]);
28. dic[sum].push(par);
29. }
30. **if** (frec\_sumas.hasOwnProperty(sum))
31. {
32. frec\_sumas[sum]++;
33. max\_frec = Math.max(max\_frec, frec\_sumas[sum]);
34. }
35. **else**
36. {
37. frec\_sumas[sum] = 1;
38. }
39. }

42. **var** ans\_pares = [];
43. **for**(**var** key **in** dic)
44. {
45. *//if(kvp.Value.Count == max\_frec)*
46. **if**(dic[key].length == max\_frec)
47. {
49. **if**(key > max\_sum)
50. {
51. ans\_pares = [];
52. ans\_pares = dic[key];
53. }
54. ans\_pares = dic[key];
55. }
56. }
58. **if** (ans\_pares.length <= 1) **return** [];
60. **return** ans\_pares;


64. }

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp2

{

class Program

{

static List<int[]> findPair(int[] arr1, int[] arr2)

{

Dictionary<int, List<int[]>> dic = new Dictionary<int, List<int[]>>();

Dictionary<int, int> frec\_sumas = new Dictionary<int, int>();

int sum = 0;

int max\_sum = 0;

int max\_frec = 0;

for (int i = 0; i < arr1.Length; i++)

{

sum = arr1[i] + arr2[i];

max\_sum = Math.Max(max\_sum, sum);

if (dic.ContainsKey(sum))

{

dic[sum].Add(new int[] { arr1[i], arr2[i] });

}

else

{

dic[sum] = new List<int[]>();

dic[sum].Add(new int[] { arr1[i], arr2[i] });

}

if (frec\_sumas.ContainsKey(sum))

{

frec\_sumas[sum]++;

max\_frec = Math.Max(max\_frec, frec\_sumas[sum]);

}

else

{

frec\_sumas[sum] = 1;

}

}

List<int[]> ans\_pares = new List<int[]>();

foreach(KeyValuePair<int , List<int[]>> kvp in dic)

{

if(kvp.Value.Count == max\_frec)

{

if(kvp.Key > max\_sum)

{

ans\_pares = new List<int[]>();

ans\_pares = kvp.Value;

}

ans\_pares = kvp.Value;

}

}

if (ans\_pares.Count <= 1) return new List<int[]>();

return ans\_pares;

}

static void Main(string[] args)

{

int[] arr1 = { 0, 1, 3, 4, 5, 0, 0, 0, 0, 0 };

int[] arr2 = { 1, 0, 3, 2, 1, 1, 1, 1, 1, 1 };

//int[] arr1 = { };

//int[] arr2 = { };

// Expected: '[[1, -1], [3, -3], [5, -5]]', instead got: '[]'

// int[] arr1 = { 1, 3, 5 };

//int[] arr2 = { -1, -3, -5 };

//Test.assertSimilar(findPair([1, 2, 3, 0, 5, -2],[-1, 2, -3, 4, -5, 6]) , [[2,2],[0,4],[-2,6]])

//int[] arr1 = { 1, 2, 3, 0, 5, -2 };

//int[] arr2 = { -1, 2, -3, 4, -5, 6 };

//Test.assertSimilar(findPair([1, 2, 3, 0, 5, -2],[-1, 2, -3, 4, -5, 6]) , [[2,2],[0,4],[-2,6]])

// int[] arr1 = { 1, 2, 3, 4, 5 };

//int[] arr2 = { 9, 8, 0, 0, 0 };

//int[] arr1 = { 1, 2, 3, 4, 5 };

//int[] arr2 = { 0, 0, 0, 0, 0 };

//int[] arr1 = { 1, 2, 3, 4, 5 };

//int[] arr2 = { 5, 4, 3, 2, 1 };

List<int[]> ans = findPair(arr1, arr2);

foreach (int[] array in ans)

{

foreach (int item in array)

{

Console.Write(item + " ");

}

Console.WriteLine();

}

//int[] arr = { -2, 5, -102, -16, -1, -2, -367, -9 };

//Console.WriteLine(NthSmallest(arr, 3));

Console.ReadLine();

}

}

}