**4 kyu**

**Range Extraction**

3275988% of 940289of 6,540[jhoffner](https://www.codewars.com/users/jhoffner)

C#

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

Details

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

[Forks (5)](https://www.codewars.com/kata/range-extraction/forks/csharp)

[Discourse (96)](https://www.codewars.com/kata/range-extraction/discuss/csharp)

* Add to Collection
* |
* Share this kata:

A format for expressing an ordered list of integers is to use a comma separated list of either

* individual integers
* or a range of integers denoted by the starting integer separated from the end integer in the range by a dash, '-'. The range includes all integers in the interval including both endpoints. It is not considered a range unless it spans at least 3 numbers. For example ("12, 13, 15-17")

Complete the solution so that it takes a list of integers in increasing order and returns a correctly formatted string in the range format.

**Example:**

solution([-6, -3, -2, -1, 0, 1, 3, 4, 5, 7, 8, 9, 10, 11, 14, 15, 17, 18, 19, 20]);

// returns "-6,-3-1,3-5,7-11,14,15,17-20"

<https://www.codewars.com/kata/range-extraction/csharp>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp1

{

class Program

{

public static string Extract(int[] arr)

{

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

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

bool flag = false;

int i;

for ( i = 0; i + 1 < arr.Length; i++)

{

//rango.Add(arr[i]);

if ( arr[i] + 1 == arr[i + 1])

{

if(flag == false)

{

rango.Add(arr[i]);

flag = true;

}

rango.Add(arr[i + 1]);

}

else

{

if (flag == false)

{

rango.Add(arr[i]);

}

r.Add(rango);

rango = new List<int>();

//rango.Add(arr[i]);

flag = false;

}

}

if(rango.Count > 0)

{

r.Add(rango);

}

else if (i < arr.Length)

{

rango.Add(arr[i]);

r.Add(rango);

}

List<string> ans = new List<string>();

foreach(List<int> lista in r)

{

if (lista.Count < 3)

{

foreach (int item in lista)

{

//Console.Write(item + " ");

ans.Add(item.ToString());

}

}

else

{

//concat += lista[0] + "-" + lista[lista.Count - 1];

ans.Add(lista[0] + "-" + lista[lista.Count - 1]);

}

}

return string.Join(",", ans);

}

public static string Extract(int[] args)

{

var result = new StringBuilder();

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

{

var startAt = args[i];

while (i + 1 < args.Length && args[i + 1] - args[i] == 1) ++i;

var endAt = args[i];

if (endAt == startAt)

{

result.Append(startAt + ",");

}

else if (endAt - startAt == 1)

{

result.Append($"{startAt},{endAt},");

}

else

{

result.Append($"{startAt}-{endAt},");

}

}

return result.ToString().TrimEnd(',');

}

static void Main(string[] args)

{

//solution(

//[-6,

//-3, -2, -1, 0, 1,

//3, 4, 5,

//7, 8, 9, 10, 11,

//14,

//15,

//17, 18, 19, 20]);

// returns "-6,-3-1,3-5,7-11,14,15,17-20"

int[] arr = { -6, -3, -2, -1, 0, 1, 3, 4, 5, 7, 8, 9, 10, 11, 14, 15, 17, 18, 19, 20 };

Console.WriteLine( Extract(arr));

Console.ReadLine();

}

}

}