**Looking for a benefactor**

37379% *of* 19999 *of* 1,515[g964](https://www.codewars.com/users/g964)

C#

* [Train Again](https://www.codewars.com/kata/looking-for-a-benefactor/train/csharp)
* [Next Kata](https://www.codewars.com/trainer/csharp)

Details

[Solutions](https://www.codewars.com/kata/looking-for-a-benefactor/solutions/csharp)

[Forks (1)](https://www.codewars.com/kata/looking-for-a-benefactor/forks/csharp)

[Discourse (49)](https://www.codewars.com/kata/looking-for-a-benefactor/discuss/csharp)

* Add to Collection
* |
* Share this kata:

The accounts of the "Fat to Fit Club (FFC)" association are supervised by John as a volunteered accountant. The association is funded through financial donations from generous benefactors. John has a list of the first n donations: [14, 30, 5, 7, 9, 11, 15] He wants to know how much the next benefactor should give to the association so that the average of the first n + 1 donations should reach an average of 30. After doing the math he found 149. He thinks that he made a mistake. Could you help him?

if dons = [14, 30, 5, 7, 9, 11, 15] then new\_avg(dons, 30) --> 149

The function new\_avg(arr, navg) should return the expected donation (rounded up to the next integer) that will permit to reach the average navg.

Should the last donation be a non positive number (<= 0) John wants us to throw an error (return Nothing in Haskell, return None in F# and Ocaml, return -1 in C) so that he clearly sees that his expectations are not great enough.

Notes:

* all donations are numbers, arr can be empty.
* See examples below to see which error is to be thrown.

new\_avg([14, 30, 5, 7, 9, 11, 15], 92) should return 645

new\_avg([14, 30, 5, 7, 9, 11, 15], 2) should raise an error (ValueError or invalid\_argument)

<https://www.codewars.com/kata/looking-for-a-benefactor/csharp>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

public static long NewAvg(double[] arr, double navg)

{

if (navg <= 0 || arr == null) throw new ArgumentException();

double sum = arr.Sum();

long x = (long)Math.Ceiling( (navg \* (arr.Length + 1)) - sum);

if (x <= 0) throw new ArgumentException();

return x;

}

static void Main(string[] args)

{

// double[] dons = {14, 30, 5, 7, 9, 11, 15};

//Console.WriteLine( NewAvg(dons, 30)); // 149

//double[] a = new double[] { 14.0, 30.0, 5.0, 7.0, 9.0, 11.0, 16.0 };

//Console.WriteLine( NewAvg(a, 90)); // , 628);

double[] a = null;

Console.WriteLine(NewAvg(a, 0));

Console.ReadLine();

}

}

}

using System;

using System.Linq;

public class NewAverage

{

public static long NewAvg(double[] arr, double navg)

{

var donation = (arr.Length + 1) \* navg - arr.Sum();

if (donation <= 0)

{

throw new ArgumentException();

}

return (long)Math.Ceiling(donation);

}

}