**6 kyu**

**Find the missing term in an Arithmetic Progression**

2974789% of 941360of 5,957[Coldberg](https://www.codewars.com/users/Coldberg)

C#

* [TRAIN AGAIN](https://www.codewars.com/kata/find-the-missing-term-in-an-arithmetic-progression/train/csharp)
* [NEXT KATA](https://www.codewars.com/trainer/csharp)

Details

[Solutions](https://www.codewars.com/kata/find-the-missing-term-in-an-arithmetic-progression/solutions/csharp)

[Forks (4)](https://www.codewars.com/kata/find-the-missing-term-in-an-arithmetic-progression/forks/csharp)

[Discourse (135)](https://www.codewars.com/kata/find-the-missing-term-in-an-arithmetic-progression/discuss/csharp)

* Add to Collection
* |
* Share this kata:

An Arithmetic Progression is defined as one in which there is a constant difference between the consecutive terms of a given series of numbers. You are provided with consecutive elements of an Arithmetic Progression. There is however one hitch: exactly one term from the original series is missing from the set of numbers which have been given to you. The rest of the given series is the same as the original AP. Find the missing term.

You have to write the function findMissing(list), list will always be at least 3 numbers. The missing term will never be the first or last one.

Example

Kata.FindMissing(new List<int> {1, 3, 5, 9, 11}) => 7

```

PS: This is a sample question of the facebook engineer challenge on interviewstreet. I found it quite fun to solve on paper using math, derive the algo that way. Have fun!

https://www.codewars.com/kata/find-the-missing-term-in-an-arithmetic-progression/train/csharp

namespace Solution

{

using NUnit.Framework;

using System;

using System.Collections.Generic;

[TestFixture]

public class Sample\_Test

{

private static IEnumerable<TestCaseData> testCases

{

get

{

yield return new TestCaseData(new[] {new List<int> {1, 3, 5, 9, 11}}).Returns(7);

yield return new TestCaseData(new[] {new List<int> {0, 5, 10, 20, 25}}).Returns(15);

yield return new TestCaseData(new[] {new List<int> {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11}}).Returns(10);

yield return new TestCaseData(new[] {new List<int> {1040, 1220, 1580}}).Returns(1400);

}

}

[Test, TestCaseSource("testCases")]

public int Test(List<int> list) => Kata.FindMissing(list);

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp1

{

class Program

{

public static int FindMissing(List<int> list)

{

int a = list[1] - list[0];

int b = list[list.Count - 1] - list[list.Count - 2];

int min\_dif = Math.Min(a, b);

for (int i = 1; i < list.Count; i++)

{

if(list[i-1]+min\_dif != list[i])

{

return list[i - 1] + min\_dif;

}

}

return -1;

}

static void Main(string[] args)

{

List<int> lista = new List<int>(new int[] { 1040, 1220, 1580 });

Console.WriteLine( FindMissing(lista));

Console.ReadLine();

}

}

}