Define the *permutation element shift* as the difference between the element's value and its index. For example, for permutation [1, 0, 2, 3] of array [0, 1, 2, 3] shifts for the respective elements are [1, -1, 0, 0].

Define the *permutation shift* as the difference between the maximal and the minimal *element shifts*among all of the permutation elements.

Given a permutation of integers from 0 to n - 1 for some integer n, find its shift.

**Example**

For permutation = [1, 0, 2, 3], the output should be  
permutationShift(permutation) = 2.

**Input/Output**

* **[time limit] 3000ms (cs)**
* **[input] array.integer permutation**

Array of integers containing each integer from 0to n - 1, inclusive, exactly once (where n is the array length).

*Guaranteed constraints:*  
1 ≤ permutation.length ≤ 10,

* **[output] integer**

**[C#] Syntax Tips**

// Prints help message to the console

// Returns a string

string helloWorld(string name) {

Console.Write("This prints to the console when you Run Tests");

return "Hello, " + name;

}

<https://codefights.com/challenge/ySFaTjaFLDnKb39iF/solutions>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static int permutationShift(int[] permutation)

{

int min = int.MaxValue;

int max = int.MinValue;

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

{

min = Math.Min(min, permutation[i] - i);

max = Math.Max(max, permutation[i] - i);

}

return max - min;

}

static void Main(string[] args)

{

int[] permutation = { 5, 4, 3, 2, 1, 0 };

Console.WriteLine(permutationShift(permutation));

Console.ReadLine();

}

}

}