Let's define *digit degree* of some positive integer as the number of times we need to replace this number with the sum of its digits until we get to a one digit number.

Given an integer, find its digit degree.

Example

* For n = 5, the output should be  
  digitDegree(n) = 0;
* For n = 100, the output should be  
  digitDegree(n) = 1.  
  1 + 0 + 0 = 1.
* For n = 91, the output should be  
  digitDegree(n) = 2.  
  9 + 1 = 10 -> 1 + 0 = 1.

Input/Output

* **[execution time limit] 3 seconds (cs)**
* **[input] integer n**

*Guaranteed constraints:*  
5 ≤ n ≤ 109.

* **[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://app.codesignal.com/challenge/KoZk8DcvWAeeHfmyu>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace ConsoleApp1

{

class Program

{

static int Sum(int n )

{

int suma = 0;

while(n > 0)

{

suma += (n % 10);

n /= 10;

}

return suma;

}

static int digitDegree(int n)

{

int ans = 0;

while(n.ToString().Length > 1)

{

ans++;

n = Sum(n);

}

return ans;

}

static void Main(string[] args)

{

int n = 91;

Console.WriteLine(digitDegree(n));

Console.ReadLine();

}

}

}