Find the last digit of n!([factorial](keyword://factorial)), which is different from zero.

**Example**

* For n = 5, the output should be  
  lastDigitDiffZero(n) = 2.  
  5! = 1 · 2 · 3 · 4 · 5 = 120.
* For n = 6, the output should be  
  lastDigitDiffZero(n) = 2.  
  6! =1 · 2 · 3 · 4 · 5 · 6 = 720.
* For n = 10, the output should be  
  lastDigitDiffZero(n) = 8.  
  10! = 3628800.
* **[execution time limit] 3 seconds (cs)**
* **[input] integer64 n**

*Guaranteed constraints:*  
1 ≤ 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://codefights.com/challenge/G77e49Sztk9BY8GA2/solutions>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp28

{

class Program

{

static int lastDigitDiffZero(long n)

{

int[] dig = { 1, 1, 2, 6, 4, 2, 2, 4, 2, 8 };

if (n < 10)

return dig[n];

// Check whether tens (or second last) digit

// is odd or even

// If n = 375, So n/10 = 37 and (n/10)%10 = 7

// Applying formula for even and odd cases.

if (((n / 10) % 10) % 2 == 0)

return (6 \* lastDigitDiffZero(n / 5) \* dig[n % 10]) % 10;

else

return (4 \* lastDigitDiffZero(n / 5) \* dig[n % 10]) % 10;

}

int lastNon0Digit(int n)

{

int[] dig = { 1, 1, 2, 6, 4, 2, 2, 4, 2, 8 };

if (n < 10)

return dig[n];

// Check whether tens (or second last) digit

// is odd or even

// If n = 375, So n/10 = 37 and (n/10)%10 = 7

// Applying formula for even and odd cases.

if (((n / 10) % 10) % 2 == 0)

return (6 \* lastNon0Digit(n / 5) \* dig[n % 10]) % 10;

else

return (4 \* lastNon0Digit(n / 5) \* dig[n % 10]) % 10;

}

static void Main(string[] args)

{

}

}

}