Given an integer num, write an algorithm that determines if the given number has consecutive 1s in its binary representation.

**Example**

* For num = 3, the output should be  
  consecutiveBit(num) = true.

310 is 112, which contains a pair of consecutive 1s.

* For num = 21, the output should be  
  consecutiveBit(num) = false.

2110 is 101012, which contains no consecutive 1s.

**Input/Output**

* **[time limit] 6000ms (cs)**
* **[input] integer num**

*Constraints:*  
0 ≤ num < 231.

* **[output] boolean**

Return true if the boolean representation of numcontains consecutive 1s, otherwise return false.

<https://codefights.com/challenge/fW66dxr49QERhv8KK?utm_source=emailNotification&utm_medium=email&utm_campaign=featuredChallenge>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static bool consecutiveBit(int num)

{

bool ant = false;

while (num > 0)

{

if (num % 2 == 1)

{

if (ant) return true;

ant = true;

}

else

{

ant = false;

}

num /= 2;

}

return false;

}

static void Main(string[] args)

{

Console.WriteLine(consecutiveBit(3));

Console.ReadLine();

}

}

}

----OTRA SOLUCION--------

static bool consecutiveBit(int num)

{

return Convert.ToString(num, 2).IndexOf("11") > -1 ? true : false;

}