You're given two integers, n and m. Find position of the rightmost bit in which they differ in their binary representations (it is guaranteed that such a bit exists), counting from right to left.

Return the value of 2position\_of\_the\_found\_bit (0-based).

**Example**

For n = 11 and m = 13, the output should be  
differentRightmostBit(n, m) = 2.

1110 = 10**1**12, 1310 = 11**0**12, the rightmost bit in which they differ is the bit at position 1 (0-based) from the right in the binary representations.  
So the answer is 21 = 2.

**Input/Output**

* **[time limit] 3000ms (cs)**
* **[input] integer n**

*Constraints:*  
0 ≤ n ≤ 230.

* **[input] integer m**

*Constraints:*  
0 ≤ m ≤ 230,  
n ≠ m.

* **[output] integer**

<https://codefights.com/arcade/code-arcade/corner-of-0s-and-1s/whz5JzszYTdXW6aNA>

<https://wxtp.wordpress.com/2016/09/26/how-to-solve-differentrightmostbit-in-codefights/>

int differentRightmostBit(int n, int m)

{

return -~((~(n ^ m)) ^ ((~(n ^ m)) + 1)) / 2;

}