Presented with the integer n, find the 0-based position of the second rightmost zero bit in its binary representation (it is guaranteed that such a bit exists), counting from right to left.

Return the value of 2position\_of\_the\_found\_bit.

**Example**

For n = 37, the output should be  
secondRightmostZeroBit(n) = 8.

3710 = 10**0**1012. The second rightmost zero bit is at position 3 (0-based) from the right in the binary representation of n.  
Thus, the answer is 23 = 8.

**Input/Output**

* **[time limit] 4000ms (py)**
* **[input] integer n**

*Constraints:*  
4 ≤ n ≤ 230.

* **[output] integer**

<https://codefights.com/arcade/code-arcade/corner-of-0s-and-1s/9nSj6DgqLDsBePJha>

<https://loctv.wordpress.com/2016/09/25/problem-find-the-second-rightmost-bit-in-a-number-1-line-code/>

def secondRightmostZeroBit(n):

    return 2\*\*(len(bin(n)[2:])-1-bin(n)[2:].rfind('0',0,len(bin(n)[2:])-(len(bin(n)[2:]) - bin(n)[2:].rfind('0'))))