 将2的幂次方写成二进制形式后，很容易就会发现有一个特点：二进制中只有一个1，并且1后面跟了n个0； 因此问题可以转化为判断1后面是否跟了n个0就可以了。

        如果将这个数减去1后会发现，仅有的那个1会变为0，而原来的那n个0会变为1；因此将原来的数与去减去1后的数字进行与运算后会发现为零。

即：十进制 二进制

1 1

2 10

4 100

8 1000

16 10000

3 11

7 111

15 1111

       最快速的方法：

      (number & number - 1) == 0

      原因：因为2的N次方换算是二进制为10……0这样的形式(0除外)。与上自己-1的位数，这们得到结果为0。例如。8的二进制为1000；8-1=7，7的二进制为111。两者相与的结果为0。计算如下：  
         1000  
     & 0111  
        -------  
        0000

**class** **Solution** {

**public**:

bool isPowerOfTwo(**int** n) {

**if**(n<=0) **return** **false**;

**return** (n&(n-1))==0;

}

};