1. 题目
2. 题干

给你一个整数 n ，请你将 1 到 n 的二进制表示连接起来，并返回连接结果对应的 **十进制** 数字对 109 + 7 取余的结果。

1. 示例

示例 1：

输入：n = 1

输出：1

解释：二进制的 "1" 对应着十进制的 1 。

示例 2：

输入：n = 3

输出：27

解释：二进制下，1，2 和 3 分别对应 "1" ，"10" 和 "11" 。

将它们依次连接，我们得到 "11011" ，对应着十进制的 27 。

示例 3：

输入：n = 12

输出：505379714

解释：连接结果为 "1101110010111011110001001101010111100" 。

对应的十进制数字为 118505380540 。

对 109 + 7 取余后，结果为 505379714 。

提示：

1 <= n <= 105

1. 题解
2. 思路

从题目上来看，这是一个十进制转二进制，再转十进制的问题，如果按照这种思路硬做的话会比较麻烦。再仔细分析，可以得出这是一个二进制的左移问题，给出整数n，从1开始连接二进制数一直到n，我们发现，已经连接过的数在这个过程中会不断地左移，每次左移的位数都是当前整数的二进制位数， 然后加上当前的整数。因此，可以得出递推公式：

res = res << 二进制位数(i) + I;

因此，可以从1开始迭代到n，得出最终的结果，不过要注意的是这种数值比较大的运算中要时刻警惕溢出的问题，首先，我们需要设定res结果为Long类型，同时在每个迭代中也要及时地取模防止溢出问题出现。

1. 代码实现

class Solution {

    public int concatenatedBinary(int n) {

        //获取每个数字转换为二进制的长度

        int mod = 1000000007;

        //二进制移位

        //当前的结果左移二进制的位数 + 当前的数

        long res = 0;

        for(int i = 1; i <= n; ++i){

            res = ((res << ten2two(i)) + i) % mod;

        }

        return (int)res;

    }

    public int ten2two(int ten){

        StringBuilder out = new StringBuilder();

        while(ten != 0){

            out.append(ten % 2);

            ten /= 2;

        }

        return out.length();

    }

}