# 题目

给定一个正整数 n，找到并返回 n 的二进制表示中两个 相邻 1 之间的 最长距离 。如果不存在两个相邻的 1，返回 0 。

如果只有 0 将两个 1 分隔开（可能不存在 0 ），则认为这两个 1 彼此 相邻 。两个 1 之间的距离是它们的二进制表示中位置的绝对差。例如，"1001" 中的两个 1 的距离为 3 。

示例 1：

输入：n = 22

输出：2

解释：22 的二进制是 "10110" 。

在 22 的二进制表示中，有三个 1，组成两对相邻的 1 。

第一对相邻的 1 中，两个 1 之间的距离为 2 。

第二对相邻的 1 中，两个 1 之间的距离为 1 。

答案取两个距离之中最大的，也就是 2 。

示例 2：

输入：n = 8

输出：0

解释：8 的二进制是 "1000" 。

在 8 的二进制表示中没有相邻的两个 1，所以返回 0 。

示例 3：

输入：n = 5

输出：2

解释：5 的二进制是 "101" 。

提示：

1 <= n <= 109

# 分析

## 方法一：位运算

思路：

我们可以使用一个循环从n二进制表示的低位开始进行遍历，并找出所有的1。我们用一个变量last记录上一个找到的1的位置。如果当前在第i位找到了1，那么就用i−last更新答案，再将last更新为i即可。

在循环的每一步中，我们可以使用位运算n & 1获取n的最低位，判断其是否为1。在这之后，我们将n右移一位：n = n >> 1，这样在第i步时，n & 1得到的就是初始n的第i个二进制位。

代码：

class Solution {

public:

int binaryGap(int n) {

int last = -1;

int ans = 0;

for (int i = 0; n; i++) {

// 判断是否为1

if (n & 1) {

if (last != -1) {

// 更新最后一个1的位置

ans = max(ans, i - last);

}

last = i;

}

n = n >> 1;

}

return ans;

}

};