二进制间距

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 = 5

输出： 2

解释：

5 的二进制是 "101" 。

示例 3：

输入： n = 6

输出： 1

解释：

6 的二进制是 "110" 。

示例 4：

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 5：

输入： n = 1

输出： 0

```
class Solution {
public:
    int binaryGap(int n) {
        /*
        若 N = 22 = 10110。则我们会记下 A = [1, 2, 4]。则我们可以在数组中计算我们的答案
        */
        int num[32];
        int cout=0;
        for(int i=0;i<32;i++)
        {
            if((n>>i)&1==1)
            {
                num[cout++]=i;//记录为 1 的索引下标
            }
        }
        int res=0;
        for(int i=0;i<cout-1;i++)
        {
            res=max(res,num[i+1]-num[i]);
        }
        return res;
    }
};
```