899 Binary Gap (link)

Description

Given a positive integer `n`, find and return *the **longest distance** between any two **adjacent** `1`'s in the binary representation of `n`. If there are no two adjacent `1`'s, return `0`.*

Two `1`'s are **adjacent** if there are only `0`'s separating them (possibly no `0`'s). The **distance** between two `1`'s is the absolute difference between their bit positions. For example, the two `1`'s in `"1001"` have a distance of 3.

Example 1:

```
Input: n = 22
Output: 2
Explanation: 22 in binary is "10110".
The first adjacent pair of 1's is "10110" with a distance of 2.
The second adjacent pair of 1's is "10110" with a distance of 1.
The answer is the largest of these two distances, which is 2.
Note that "10110" is not a valid pair since there is a 1 separating the two 1's under
```

Example 2:

```
Input: n = 8
Output: 0
Explanation: 8 in binary is "1000".
There are not any adjacent pairs of 1's in the binary representation of 8, so we retu
```

Example 3:

```
Input: n = 5
Output: 2
Explanation: 5 in binary is "101".
```

Constraints:

- `1 <= n <= 10^9`

(scroll down for solution)

Solution

Language: cpp

Status: Accepted

```cpp
class Solution {
public:
    int binaryGap(int n) {
        int maxDistance = 0;
        int prevOnePos = -1;
        int currentPos = 0;

        while (n > 0) {
            if (n & 1) {
                if (prevOnePos != -1) {
                    maxDistance = max(maxDistance, currentPos - prevOnePos);
                }
                prevOnePos = currentPos;
            }
            n >>= 1;
            currentPos++;
        }

        return maxDistance;
    }
};
```