

# Problem 1545: Find Kth Bit in Nth Binary String

## Problem Information

**Difficulty:** Medium

**Acceptance Rate:** 70.14%

**Paid Only:** No

**Tags:** String, Recursion, Simulation

## Problem Description

Given two positive integers  $n$  and  $k$ , the binary string  $S_n$  is formed as follows:

$S_1 = "0"$   $S_i = S_{i-1} + "1" + \text{reverse}(\text{invert}(S_{i-1}))$  for  $i > 1$

Where  $+$  denotes the concatenation operation,  $\text{reverse}(x)$  returns the reversed string  $x$ , and  $\text{invert}(x)$  inverts all the bits in  $x$  ( $0$  changes to  $1$  and  $1$  changes to  $0$ ).

For example, the first four strings in the above sequence are:

$S_1 = "0"$   $S_2 = "011"$   $S_3 = "0111001"$   $S_4 = "011100110110001"$

Return the  $k$ th bit in  $S_n$ . It is guaranteed that  $k$  is valid for the given  $n$ .

**Example 1:**

**Input:**  $n = 3, k = 1$  **Output:** "0" **Explanation:**  $S_3$  is "0111001". The 1st bit is "0".

**Example 2:**

**Input:**  $n = 4, k = 11$  **Output:** "1" **Explanation:**  $S_4$  is "011100110110001". The 11th bit is "1".

**Constraints:**

\*`1 <= n <= 20` \*`1 <= k <= 2n - 1`

## Code Snippets

### C++:

```
class Solution {  
public:  
    char findKthBit(int n, int k) {  
  
    }  
};
```

### Java:

```
class Solution {  
    public char findKthBit(int n, int k) {  
  
    }  
}
```

### Python3:

```
class Solution:  
    def findKthBit(self, n: int, k: int) -> str:
```