5484. Find Kth Bit in Nth Binary String

My Submissions (/contest/weekly-contest-201/problems/find-kth-bit-in-nth-binary-string/submissions/)

Back to Contest (/contest/weekly-contest-201/)

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" + reverse(invert(S_{i-1})) for `i > 1`

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

For example, the first 4 strings in the above sequence are:

- S_1 = "0"
- S_2 = "0**1**1"
- S_3 = "011**1**001"
- S_4 = "0111001**1**0110001"

Return *the* k^{th} *bit in* S_n. It is guaranteed that `k` is valid for the given `n`.

User Accepted:	0
User Tried:	0
Total Accepted:	0
Total Submissions:	0
Difficulty:	Medium

Example 1:

Input: n = 3, k = 1

Output: "0"

Explanation: S_3 is "**0**111001". The first bit is "0".

Example 2:

Input: n = 4, k = 11

Output: "1"

Explanation: S_4 is "0111001101**1**0001". The 11th bit is "1".

Example 3:

Input: n = 1, k = 1

Output: "0"

Example 4:

Input: n = 2, k = 3

Output: "1"

Constraints:

- `1 <= n <= 20`
- $1 <= k <= 2^n - 1$

JavaScript ▾

```
/**
 * @param {number} n
 * @param {number} k
 * @return {character}
 */
var findKthBit = function(n, k) {

};
```

□ **Custom Testcase** Use Example Testcases

Run Submit

Help Center (/support/) | Terms (/terms/) | Privacy Policy (/privacy/)

United States (/region/)