1415. The k-th Lexicographical String of All Happy Strings of Length n

Medium | Topics | Companies | Hint

A **happy string** is a string that:

- consists only of letters of the set `['a', 'b', 'c']`.
- `s[i] != s[i + 1]` for all values of `i` from `1` to `s.length - 1` (string is 1-indexed).

For example, strings **"abc", "ac", "b"** and **"abcbabcbcb"** are all happy strings and strings **"aa", "baa"** and **"ababbc"** are not happy strings.

Given two integers `n` and `k`, consider a list of all happy strings of length `n` sorted in lexicographical order.

Return *the kth string* of this list or return an **empty string** if there are less than `k` happy strings of length `n`.

Example 1:

```
Input: n = 1, k = 3
Output: "c"
Explanation: The list ["a", "b", "c"] contains all happy strings of length 1. The third string is "c".
```

Example 2:

```
Input: n = 1, k = 4
Output: ""
Explanation: There are only 3 happy strings of length 1.
```

Example 3:

```
Input: n = 3, k = 9
Output: "cab"
Explanation: There are 12 different happy string of length 3 ["aba", "abc", "aca", "acb", "bab", "bac", "bca", "bcb", "cab", "cac", "cba", "cbc"]. You will find the 9th string = "cab"
```

Python3 | Auto

```python
class Solution:
    def getHappyString(self, n: int, k: int) -> str:
        total = 3 * (2 ** (n-1))
        if k > total:
            return ""

        res = []
        chars = {'a', 'b', 'c'}
        prev = ''

        for i in range(n):
            if i == 0:
                if k <= total // 3:
                    res.append('a')
                    prev = 'a'
                elif k <= total * 2 // 3:
                    res.append('b')
                    prev = 'b'
                else:
                    res.append('c')
                    prev = 'c'
            else:
```

Ln 23, Col 17 | Saved

Testcase | Test Result

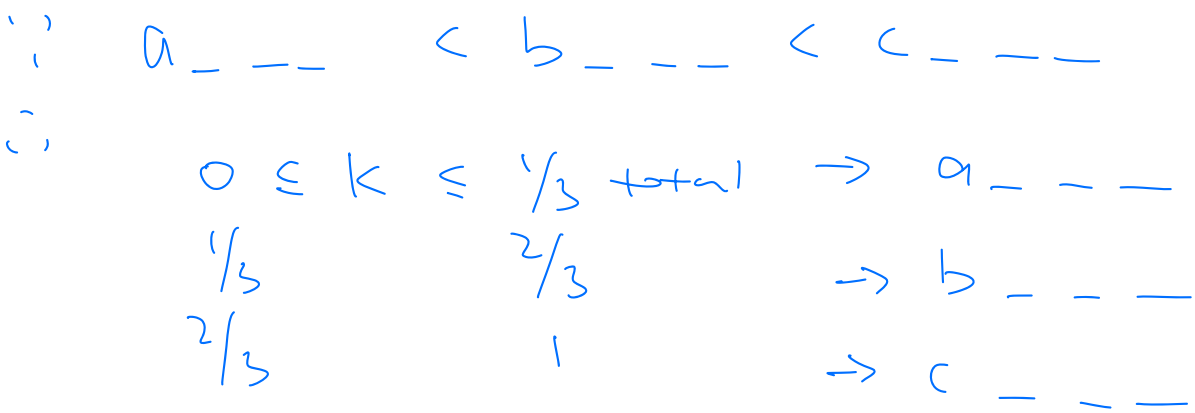
Case 1 | Case 2 | Case 3 +

n =

1

k =

3



same for the next digit. just need to be careful with the prev char, no repeat.

Accepted 345 / 345 testcases passed

AndrewC275 submitted at Feb 19, 2025 10:30

Editorial | Solution

Runtime

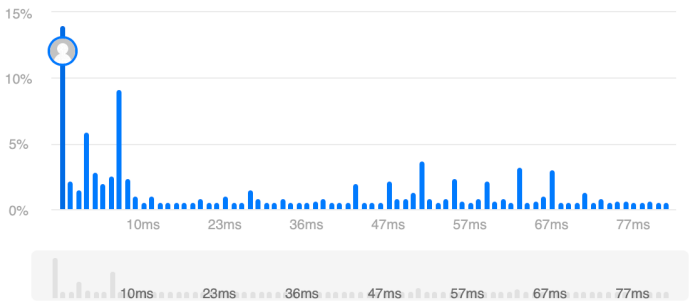
0 ms | Beats **100.00%**

Analyze Complexity

Memory

17.78 MB | Beats **82.93%**

Analyze Complexity




```python
class Solution:
    def getHappyString(self, n: int, k: int) -> str:
        total = 3 * (2 ** (n-1))
        if k > total:
            return ""

        res = []
        chars = "abc"
        left, right = 1, total

        for i in range(n):
            curr = left
            partition_size =(right - left + 1) // len(chars)

            for c in chars:
                if k in range(curr, curr + partition_size):
                    res.append(c)
                    left = curr
                    right = curr + partition_size - 1
                    chars = "abc".replace(c, "")
                    break
                curr += partition_size

        return "".join(res)
```