跳跃游戏 VII

给你一个下标从 **0** 开始的二进制字符串 s 和两个整数 minJump 和 maxJump 。一开始，你在下标 0 处，且该位置的值一定为 '0' 。当同时满足如下条件时，你可以从下标 i 移动到下标 j 处：

- i + minJump <= j <= min(i + maxJump, s.length - 1) 且
- s[j] == '0'.

如果你可以到达 s 的下标 s.length - 1 处，请你返回 true ，否则返回 false 。

示例 1：

输入： s = "011010", minJump = 2, maxJump = 3

输出： true

解释：

第一步，从下标 0 移动到下标 3 。

第二步，从下标 3 移动到下标 5 。

示例 2：

输入： s = "01101110", minJump = 2, maxJump = 3

输出： false

```
class Solution:
    def canReach(self, s: str, minJump: int, maxJump: int) -> bool:
        #记忆状态+滑动窗口
        n = len(s)
        #初始化状态列表
        dp = [0] * n
        dp[0] = 1
        l = r = 0
        cur = 1 #可用‘0’数量 初始 0 位置一定为‘0’ 则 cur 值为 1

        for i in range(minJump, n):
            if s[i] == '0':
                if cur:
                    dp[i] = 1 #如果当前遍历元素为‘0’ 且可用‘0’数量大于 0 状态置
1
            r += 1 #右指针主动右移
```

```
            cur += dp[r] #窗口变化 更新当前可用‘0’数量

            if i - l >= maxJump:
                cur -= dp[l] #将不满足条件的去除 更新当前可用‘0’数量
                l += 1 #左指针被动左移
        return bool(dp[-1])
```