# 题目

有两种特殊字符：

第一种字符可以用一比特 0 表示

第二种字符可以用两比特（10 或 11）表示

给你一个以 0 结尾的二进制数组 bits ，如果最后一个字符必须是一个一比特字符，则返回 true 。

示例 1:

输入: bits = [1, 0, 0]

输出: true

解释: 唯一的解码方式是将其解析为一个两比特字符和一个一比特字符。

所以最后一个字符是一比特字符。

示例 2:

输入：bits = [1,1,1,0]

输出：false

解释：唯一的解码方式是将其解析为两比特字符和两比特字符。

所以最后一个字符不是一比特字符。

提示:

1 <= bits.length <= 1000

bits[i] 为 0 或 1

# 分析

## 方法一：模拟

思路：

可以使用迭代的方法遍历数组bits，根据规则判断最后一个字符是否为一比特字符。

在遍历过程中，根据当前位置i的值来判断是一比特字符还是两比特字符：如果当前位置i为1，则表示当前为两比特字符，需要将i增加2；如果当前位置i为0，则表示当前为一比特字符，只需要将i增加1。最终判断是否为一比特字符的条件是是否遍历到了数组的倒数第二个位置，如果是，则最后一个字符必定为一比特字符，返回true；否则，最后一个字符不是一比特字符，返回false。

代码：

class Solution {

public:

bool isOneBitCharacter(vector<int>& bits) {

int i = 0;

while (i < bits.size() - 1) {

if (bits[i] == 1) {

i += 2;

} else {

i++;

}

}

return i == bits.size() - 1;

}

};