# 题目

存在一个不含 0 的 环形 数组 nums ，每个 nums[i] 都表示位于下标 i 的角色应该向前或向后移动的下标个数：

* 如果 nums[i] 是正数，向前（下标递增方向）移动 |nums[i]| 步
* 如果 nums[i] 是负数，向后（下标递减方向）移动 |nums[i]| 步

因为数组是 环形 的，所以可以假设从最后一个元素向前移动一步会到达第一个元素，而第一个元素向后移动一步会到达最后一个元素。

数组中的 循环 由长度为 k 的下标序列 seq 标识：

* 遵循上述移动规则将导致一组重复下标序列 seq[0] -> seq[1] -> ... -> seq[k - 1] -> seq[0] -> ...
* 所有 nums[seq[j]] 应当不是 全正 就是 全负
* k > 1

如果 nums 中存在循环，返回 true ；否则，返回 false 。

示例 1：

输入：nums = [2,-1,1,2,2]

输出：true

解释：存在循环，按下标 0 -> 2 -> 3 -> 0 。循环长度为 3 。

示例 2：

输入：nums = [-1,2]

输出：false

解释：按下标 1 -> 1 -> 1 ... 的运动无法构成循环，因为循环的长度为 1 。根据定义，循环的长度必须大于 1 。

示例 3:

输入：nums = [-2,1,-1,-2,-2]

输出：false

解释：按下标 1 -> 2 -> 1 -> ... 的运动无法构成循环，因为 nums[1] 是正数，而 nums[2] 是负数。

所有 nums[seq[j]] 应当不是全正就是全负。

提示：

1 <= nums.length <= 5000

-1000 <= nums[i] <= 1000

nums[i] != 0

进阶：你能设计一个时间复杂度为 O(n) 且额外空间复杂度为 O(1) 的算法吗？

# 分析