457. Circular Array Loop

Medium 408 1381 Add to List Share

You are given a **circular** array `nums` of positive and negative integers. If a number *k* at an index is positive, then move forward *k* steps. Conversely, if it's negative (*-k*), move backward *k* steps. Since the array is circular, you may assume that the last element's next element is the first element, and the first element's previous element is the last element.

Determine if there is a loop (or a cycle) in `nums`. A cycle must start and end at the same index and the cycle's length > 1. Furthermore, movements in a cycle must all follow a single direction. In other words, a cycle must not consist of both forward and backward movements.

Example 1:

```
Input: [2,-1,1,2,2]
Output: true
Explanation: There is a cycle, from index 0 -> 2 -> 3 -> 0. The cycle's length is 3.
```

Example 2:

```
Input: [-1,2]
Output: false
Explanation: The movement from index 1 -> 1 -> 1 ... is not a cycle, because the cycle's length is 1. By definition the cycle's length must be greater than 1.
```

Example 3:

Java Autocomplete

```java
class Solution {
    public boolean circularArrayLoop(int[] nums) {
        for(int i=0; i < nums.length; i++)
            if(isCycle(nums, i, nums[i] > 0))
                return true;
        return false;
    }
    //detect cycle in directed graph going in one direction
    private boolean isCycle(int[] nums, int i, boolean isPos){
        if(nums[i] == 0) return true;
        if( (nums[i] > 0) != isPos) return false;
        int n = nums.length, next = (n+ (i+nums[i]) % n) % n;
        if(next == i) return false;
        int backup = nums[i];
        nums[i] = 0;
        if(isCycle(nums, next, isPos))
           return true;
        nums[i] = backup;
        return false;
    }
}
```

Your previous code was restored from your local storage. Reset to default

Testcase | Run Code Result | Debugger

Accepted Runtime: 0 ms

Your input [2,-1,1,2,2]

Output true Diff

Expected true