题目难度：MEDIUM

给你一个按升序排序的整数数组 num（可能包含重复数字），请你将它们分割成一个或多个子序列，其中每个子序列都由连续整数组成且长度至少为 3 。

如果可以完成上述分割，则返回 true ；否则，返回 false 。

示例 1：

输入: [1,2,3,3,4,5]

输出: True

解释:

你可以分割出这样两个连续子序列 :

1, 2, 3

3, 4, 5

示例 2：

输入: [1,2,3,3,4,4,5,5]

输出: True

解释:

你可以分割出这样两个连续子序列 :

1, 2, 3, 4, 5

3, 4, 5

示例 3：

输入: [1,2,3,4,4,5]

输出: False

提示：

输入的数组长度范围为 [1, 10000]

时间击败：34%（时间不咋地）

空间击败：7%（空间炸了）

思路：Sliding Windows

先判断数组长度小小于3返回FALSE（WA点1：缺少此判断）

然后数组排序，之后取最大最小值的差

如果大于10000就返回FALSE（差值大于数组长度肯定不是连续序列）

否则遍历数组值作为建立数组下标，出现一次建立数组对应下标的值就加一

然后遍历建立数组，当出现数组值大于1时，记录下标（需要形成不止一个连续子序列）

是否记录下标状态转换（只记录最前面的不是连续子序列项）

如果后一项的值大于等于前一项，那么当前项的值减一，长度（初值0）加一继续循环

否则肯定是要到当前项截止取一个子序列（因为要与后面子序列连续，值就不能大于下一项）

如果长度小于2（无法构成子序列由连续整数组成且长度至少为3）就返回FALSE

否则当前遍历下标继承记录下标值（从第一个不是连续子序列项开始）

是否记录下标状态转换

上代码：

class Solution {

public:

int len = 0, siz = 0;

bool flag = true;

bool isPossible(vector<int>& nums) {

if(nums.size() < 3) return false;

vector<int> curr(10005, 0);

sort(nums.begin(), nums.end());

if(nums[nums.size() - 1] - nums[0] > 10000) return false;

for(int i = 0; i < nums.size(); i++) curr[nums[i] - nums[0]]++;

for(int i = 0; i < 10003; i++) {

if(flag && curr[i] > 1) {

siz = i;

flag = false;

}

if(curr[i + 1] >= curr[i]) {

curr[i]--;

len++;

}

else {

curr[i]--;

if(len < 2) return false;

i = siz;

flag = true;

len = 0;

i--;

}

}

return true;

}

};