在一个给定的数组`nums`中，总是存在一个最大元素 。

查找数组中的最大元素是否至少是数组中每个其他数字的两倍。

如果是，则返回最大元素的索引，否则返回-1。

示例 1:

输入： `nums = [3, 6, 1, 0]`
输出： `1`
解释： `6`是最大的整数，对于数组中的其他整数，
`6`大于数组中其他元素的两倍。`6`的索引是`1`，所以我们返回`1`.

示例 2:

输入： `nums = [1, 2, 3, 4]`
输出： `-1`
解释： `4`没有超过`3`的两倍大，所以我们返回 `-1`.

提示:

1. `nums` 的长度范围在`[1, 50]`.
2. 每个 `nums[i]` 的整数范围在 `[0, 99]`.

思路，标记最大和次大的数字，次大乘以2和最大判断即可，最后面添加个必定最小的元素，最大次大先指向最后一个，再动态生成。

```
class Solution {
public:
    int dominantIndex(vector<int>& nums) {
        int n=nums.size();
        nums.push_back(-1);
        int max=n,max1=n;
        for(int i=0;i<n;++i){
            if(nums[max]<nums[i]){
                max1=max;
                max=i;
            }else if(nums[max1]<nums[i]){
                max1=i;
            }
        }
```

```
        if(n==1){
            return 0;
        }
        return nums[max1]*2<=nums[max]?max:-1;
    }
};
```