# Question

In a given integer array nums, there is always exactly one largest element.

Find whether the largest element in the array is at least twice as much as every other number in the array.

If it is, return the **index** of the largest element, otherwise return -1.

**Example 1:**

**Input:** nums = [3, 6, 1, 0]

**Output:** 1

**Explanation:** 6 is the largest integer, and for every other number in the array x,

6 is more than twice as big as x. The index of value 6 is 1, so we return 1.

**Example 2:**

**Input:** nums = [1, 2, 3, 4]

**Output:** -1

**Explanation:** 4 isn't at least as big as twice the value of 3, so we return -1.

**Note:**

1. nums will have a length in the range [1, 50].
2. Every nums[i] will be an integer in the range [0, 99].

# Solution

#### **Approach #1: Linear Scan [Accepted]**

**Intuition and Algorithm**

Scan through the array to find the unique largest element m, keeping track of it's index maxIndex.

Scan through the array again. If we find some x != m with m < 2\*x, we should return -1.

Otherwise, we should return maxIndex.

|  |
| --- |
| class Solution {  public int dominantIndex(int[] nums) {  int maxIndex = 0;  for (int i = 0; i < nums.length; ++i) {  if (nums[i] > nums[maxIndex])  maxIndex = i;  }  for (int i = 0; i < nums.length; ++i) {  if (maxIndex != i && nums[maxIndex] < 2 \* nums[i])  return -1;  }  return maxIndex;  }  } |

**Complexity Analysis**

* Time Complexity: O(N)*O*(*N*) where N*N* is the length of nums.
* Space Complexity: O(1)*O*(1), the space used by our int variables.