Largest Number At Least Twice of Others

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 1

Find largest two numbers and check if twice of second largest is greater than largest or not.

```
public int dominantIndex(int[] nums) {
   int maxOne = 0, maxTwo = 0, idx = 0;
   for(int i=0; i<nums.length; i++){
       if(maxOne < nums[i]){
          maxTwo = maxOne;
          maxOne = nums[i];
          idx = i;
       }
       else if(maxTwo < nums[i]){
          maxTwo = nums[i];
      }
   }
   return (maxOne>=maxTwo*2) ? idx : -1;
}
```

written by aayushgarg original link here

Solution 2

The idea is to find the largest element and second largest element from the given nums, then compare them the way the question has described.

```
class Solution {
    public int dominantIndex(int[] nums) {
        int sec_i = -1;
        int max_i = -1;
        for (int i = 0; i < nums.length; i++) {
            if (max_i == -1 || nums[i] > nums[max_i]) {
                sec_i = max_i;
                max_i = i;
            }
            else if (sec_i == -1 || nums[i] > nums[sec_i]) {
                sec_i = i;
            }
        }
        if (nums[max_i] >= nums[sec_i] * 2) return max_i;
        return -1;
    }
}
```

written by jun1013 original link here

Solution 3

Just iterate through the array and note the highest and second highest numbers. Might as well take note of the index at the same time.

One slightly clever idea was to shuffle the highest to the second-highest whenever a new highest was found. That was a way to handle the case where there are two (or more) value tied for highest.

```
class Solution:
    def dominantIndex(self, nums):
        if len(nums) == 0: return -1

        highest = -1
        secondHighest = -1
        highestIndex = 0

        for i,n in enumerate(nums):
            if n >= highest:
                secondHighest = highest
                highest = n
                highestIndex = i
            elif n > secondHighest:
                secondHighest = n

        if highest < secondHighest*2:
            highestIndex = -1

        return highestIndex
```

written by SunnyvaleCA original link here