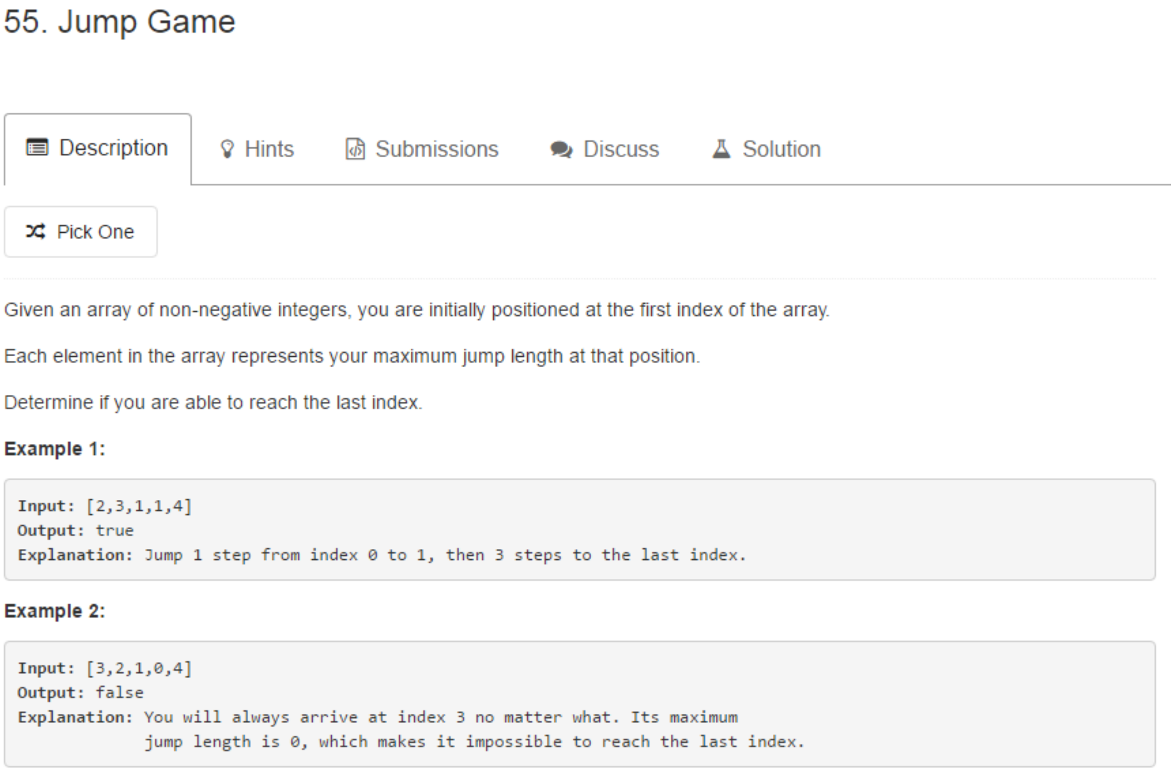


55. Jump Game

Given an array of non-negative integers, you are initially positioned at the first index of the array.

Each element in the array represents your maximum jump length at that position.

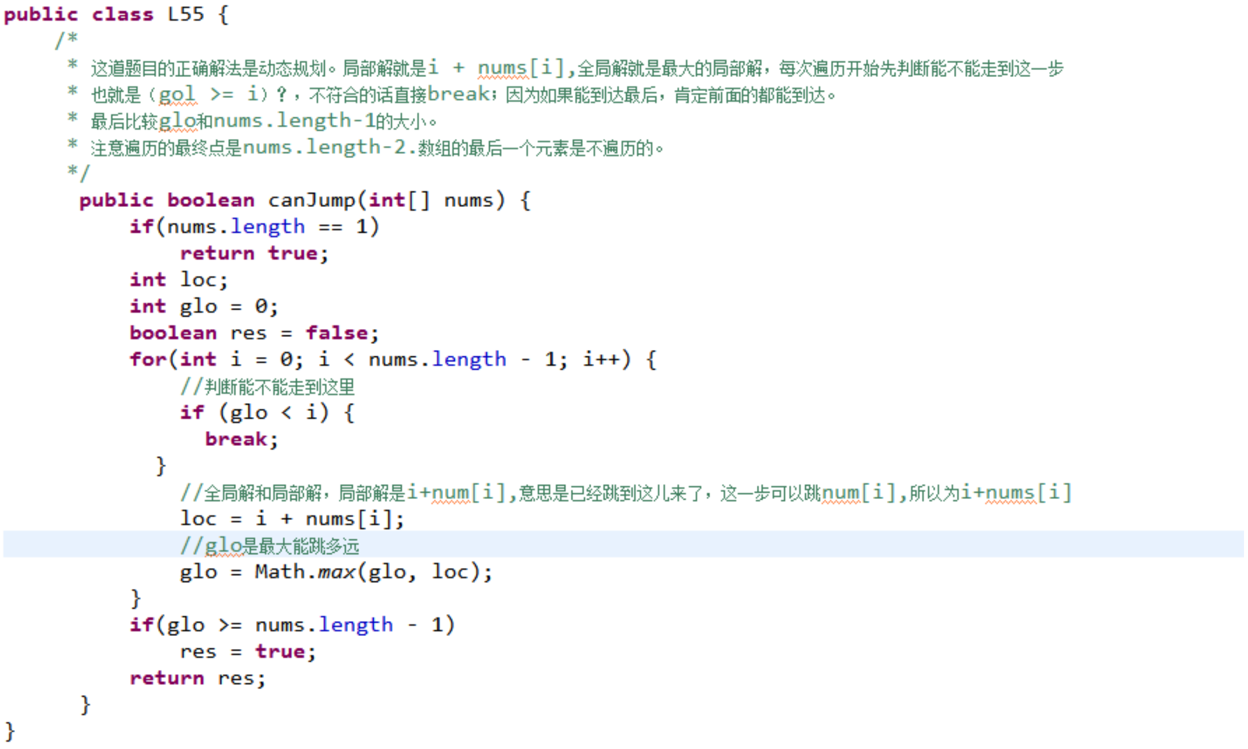
Determine if you are able to reach the last index.

Example 1:

```
Input: [2,3,1,1,4]
Output: true
Explanation: Jump 1 step from index 0 to 1, then 3 steps to the last index.
```

Example 2:

```
Input: [3,2,1,0,4]
Output: false
Explanation: You will always arrive at index 3 no matter what. Its maximum
             jump length is 0, which makes it impossible to reach the last index.
```



```
public class L55 {
    /*
     * 这道题目的正确解法是动态规划。局部解就是i + nums[i],全局解就是最大的局部解，每次遍历开始先判断能不能走到这一步
     * 也就是（gol >= i）?，不符合的话直接break；因为如果能到达最后，肯定前面的都能到达。
     * 最后比较glo和nums.length-1的大小。
     * 注意遍历的最终点是nums.length-2.数组的最后一个元素是不遍历的。
     */
      public boolean canJump(int[] nums) {
          if(nums.length == 1)
              return true;
          int loc;
          int glo = 0;
          boolean res = false;
          for(int i = 0; i < nums.length - 1; i++) {
              //判断能不能走到这里
              if (glo < i) {
                break;
            }
              //全局解和局部解，局部解是i+num[i],意思是已经跳到这儿来了，这一步可以跳num[i],所以为i+nums[i]
              loc = i + nums[i];
              //glo是最大能跳多远
              glo = Math.max(glo, loc);
          }
          if(glo >= nums.length - 1)
              res = true;
          return res;
      }
}
```