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.
```

```
class Solution {
    public boolean canJump(int[] nums) {
        if (nums == null || nums.length < 2)
            return true;
        int steps = nums[0];
        for (int i = 1; i < nums.length; i++) {
            steps--;
            if (steps < 0)
                return false;
            steps = Math.max(steps, nums[i]);
        }
        return true;
    }
}
```

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.

Your goal is to reach the last index in the minimum number of jumps.

Example:

```
Input: [2,3,1,1,4]
Output: 2
Explanation: The minimum number of jumps to reach the last
index is 2.
    Jump 1 step from index 0 to 1, then 3 steps to the last
index.
```

```
class Solution {
    public int jump(int[] nums) {
        int[] dp = new int[nums.length];
        Arrays.fill(dp, Integer.MAX_VALUE);
        dp[0] = 0;
        for (int i = 0; i < nums.length; i++) {
            for (int j = i+1; j < nums.length && j <= i + nums[i]; j++) {
                dp[j] = Math.min(dp[j], 1 + dp[i]);
            }
        }
        return dp[nums.length-1];
    }
}
```