Jump Game II Documentation

Problem Description

You are given a 0-indexed array of integers `nums` of length `n`. You are initially positioned at `nums[0]`. Each element `nums[i]` represents the maximum length of a forward jump from index `i`. In other words, if you are at `nums[i]`, you can jump to any `nums[i + j]` where:

- 0 <= j <= nums[i]
- i + j < n

Return the minimum number of jumps to reach `nums[n - 1]`. The test cases are generated such that you can reach `nums[n - 1]`.

Examples

Example 1:

Input: `nums = [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.

Example 2:

Input: `nums = [2,3,0,1,4]`

Output: `2`

Constraints

- 1 <= nums.length <= 10^4
- 0 <= nums[i] <= 1000
- It's guaranteed that you can reach `nums[n - 1]`.

Approach

1. Initialize `max_reach` to `nums[0]`, `steps` to `nums[0]`, and `jumps` to `1`.
2. Iterate over the elements of the array from index 1 to the end:
    a. Update `max_reach` to the maximum of `max_reach` and `i + nums[i]`.
    b. Decrement `steps` by 1.
    c. If `steps` becomes 0, increment `jumps` and update `steps` to `max_reach - i`.
    d. If the loop reaches the last index, return `jumps`.
3. If the loop ends without reaching the last index, return `jumps`.

Time Complexity

- The time complexity of the solution is O(n), where n is the length of the input array `nums`. This is because we traverse the array only once.