**JUMP GAME – II**

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] and
* 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].

**CODE**

class Solution {

public:

    int jump(vector<int>& nums)

    {

        int jumps = 0;

        int currentEnd = 0;

        int farthest = 0;

        for (int i = 0; i < nums.size() - 1; i++)

        {

            farthest = max(farthest, i + nums[i]);

            if (i == currentEnd) {

                jumps++;

                currentEnd = farthest;

            }

        }

        return jumps;

    }

};

**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