Documentation: Jump Game

Problem Statement

Given an integer array `nums`, where each element represents the maximum jump length at that position, determine whether it's possible to reach the last index starting from the first index.

Function Signature

def canJump(nums: List[int]) -> bool:

Determines whether it's possible to reach the last index starting from the first index.

Parameters:

nums (List[int]): An integer array where each element represents the maximum jump length at that position.

Returns:

bool: True if reaching the last index is possible, False otherwise.

Example

Example 1

Input: nums = [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: nums = [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.

Constraints

- The length of `nums` is between 1 and 10^4 (inclusive).
- Each element in `nums` is between 0 and 10^5 (inclusive).

Approach

- We traverse the array while keeping track of the maximum reachable index (`max_reachable`) from the current position.
- If at any point the current index `i` is greater than `max_reachable`, it means we cannot reach the last index.
- Update `max_reachable` by taking the maximum between its current value and `i + nums[i]`.
- If `max_reachable` is greater than or equal to the last index, return True, indicating reaching the last index is possible.
- If the loop ends and we haven't reached the last index, return False.

Example usage:

```
solution = Solution()

print(solution.canJump([2,3,1,1,4]))  # Output: True

print(solution.canJump([3,2,1,0,4]))  # Output: False
```