Find Minimum in Rotated Sorted Array

Idea:

Because the array was originally strictly increasing and then rotated, it consists of two increasing segments. The minimum is the rotation pivot. Compare the middle element to the rightmost element to decide which half is unsorted (contains the pivot):

- Set lo = hi, hi = n-1
- While lo < hi
  - mid = lo + (hi - lo)/2
  - If nums[mid] > nums[hi] → the min is to the right of mid (the right half is the rotated/unsorted one). Set lo = mid+1.
  - Else (nums[mid] < nums[hi]) → the min is at mid or to the left. Set hi = mid.
- Return nums[lo] (or nums[hi]; they're equal at the end).

Why this works: in a strictly increasing array, if nums[mid] > nums[hi], you must be in the left (higher) segment; otherwise you're in the right (lower) segment that includes the minimum.

Complexity:

- Time: $O(\log n)$
- Space: $O(1)$