**2.6 BINARY SEARCH FOR PEAK ELEMENT.**

**AIM**

To design an algorithm that finds the index of a peak element in an array using binary search in O(log n) time.

**ALGORITHM**

1. Let low = 0, high = n-1.

2. While low < high:

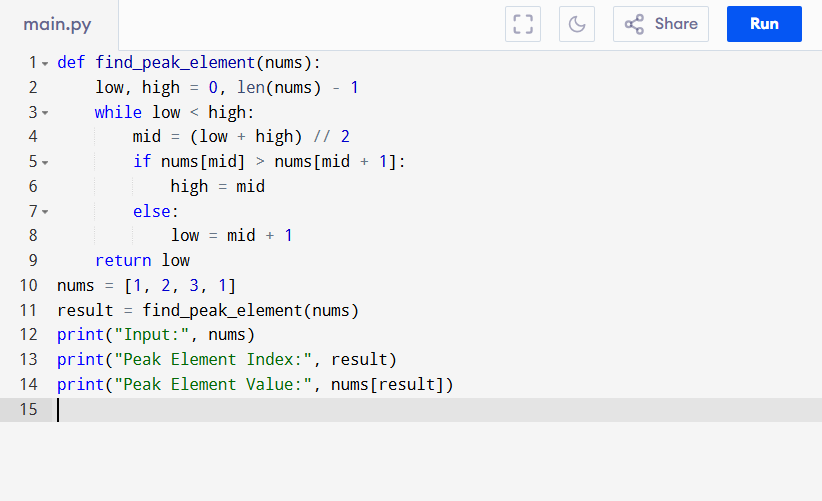
* Compute mid = (low + high) // 2.
* If nums[mid] > nums[mid + 1]:
* Then the peak lies in the left half (including mid), so set high = mid.

Else:

The peak lies in the right half, so set low = mid + 1.

3. When the loop ends, low == high and that index is a peak.

**PROGRAM**



Input:

[1, 2, 3, 1]

Output:

A screenshot of a computer program

AI-generated content may be incorrect.

**RESULT:**

Thus the program is successfully executed and the output is verified.

**PERFORMANCE ANALYSIS:**

* Time Complexity:
  + Each step reduces the search space by half → O(log n).
* Space Complexity:
  + O(1) (binary search done in-place, no extra memory).
  + Best Case: Found in first comparison → O(1).
  + Worst Case: O(log n) comparisons until narrowing down to one element.