SMI
UNIVERSITY

NAME : SAMEER KUMAR

ROLL NO : BIT-24S-005

SUBJECT : ARTIFICIAL INTELLIGENCE

TOPIC : FIND PEAK ELEMENT

DIFFICULTY : MEDIUM

TEACHER : AQSA UMAR

LEETCODE PROBLEM(162)

(EXAMPLE NO 1)

Input: nums = [1,2,3,1]

Output: 2

Explanation: 3 is a peak element and your function should return the index number 2.

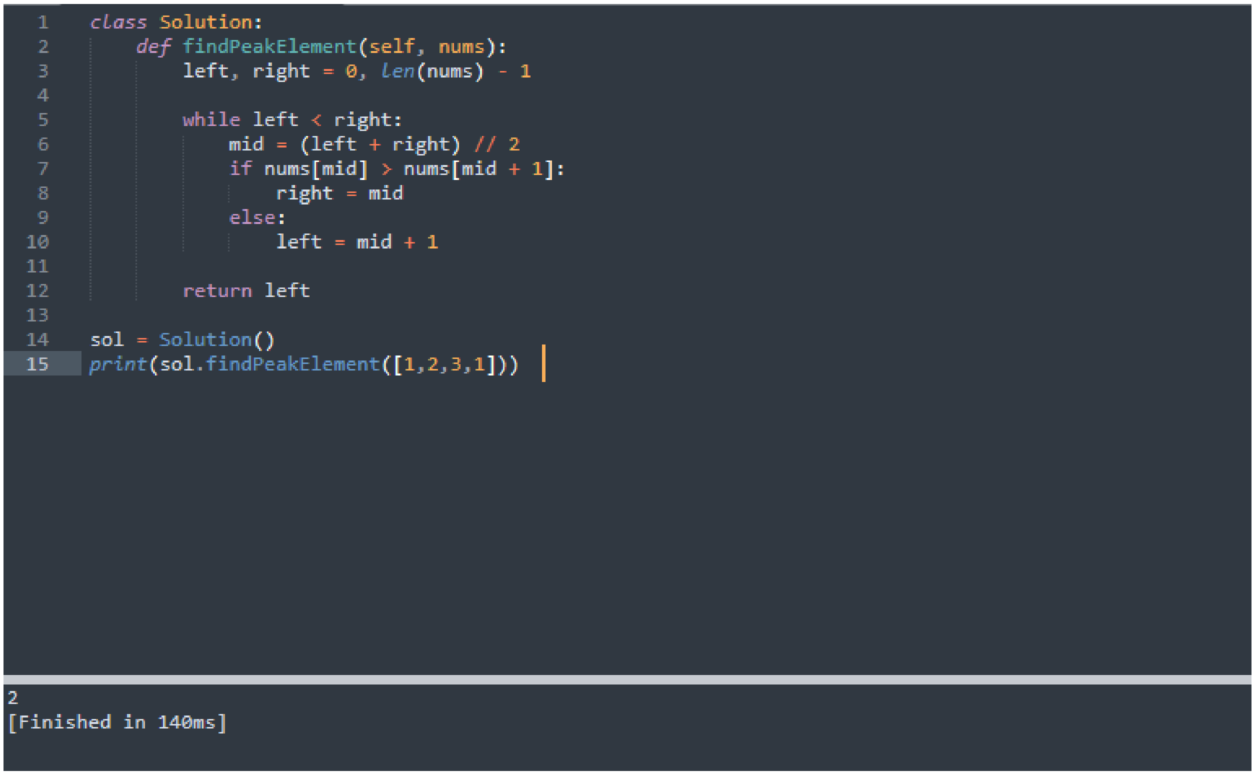
(EXAMPLE NO 2)

Example 2:

Input: nums = [1,2,1,3,5,6,4]

Output: 5

Explanation: Your function can return either index number 1 where the peak element is 2, or index number 5 where the peak element is 6.



```
class Solution:
    def findPeakElement(self, nums):
        left, right = 0, len(nums) - 1

        while left < right:
            mid = (left + right) // 2
            if nums[mid] > nums[mid + 1]:
                right = mid
            else:
                left = mid + 1

        return left

sol = Solution()
print(sol.findPeakElement([1,2,3,1]))
```

```
2
[Finished in 140ms]
```

Code Explanation :

1. **Initialization: The binary search starts with left set to 0 and right set to the last index of the array.**
2. **Binary Search Loop: The loop continues as long as left is less than right:**
   - ➢ **Middle Index Calculation: The middle index mid is calculated as the average of left and right.**
   - ➢ **Comparison with Next Element: If the element at mid is greater than the next element (nums[mid] > nums[mid + 1]), it means there is a peak in the left half (including mid), so right is set to mid.**

- ➢ **Else Condition: If the element at mid is not greater than the next element, the peak must be in the right half, so left is set to mid + 1.**

3. **Termination: When the loop exits, left equals right and points to a peak element. This is because the binary search narrows down the search space to a single element that is greater than its neighbors or is a boundary element that meets the peak condition.**