

Leetcode problem number :04

Statement:-

Given two sorted arrays nums1 and nums2 of size m and n respectively, return **the median** of the two sorted arrays.

The overall run time complexity should be $O(\log (m+n))$.

Example 1:

Input: nums1 = [1,3], nums2 = [2]

Output: 2.00000

Explanation: merged array = [1,2,3] and median is 2.

Example 2:

Input: nums1 = [1,2], nums2 = [3,4]

Output: 2.50000

Explanation: merged array = [1,2,3,4] and median is $(2 + 3) / 2 = 2.5$.

Constraints:

- `nums1.length == m`
- `nums2.length == n`
- $0 \leq m \leq 1000$
- $0 \leq n \leq 1000$
- $1 \leq m + n \leq 2000$
- $-10^6 \leq \text{nums1}[i], \text{nums2}[i] \leq 10^6$

Explanation

आपको दो **sorted arrays** दिए गए हैं:

- `nums1` (आकार = `m`)
- `nums2` (आकार = `n`)

दोनों arrays पहले से बढ़ते क्रम (**ascending order**) में हैं।

👉 **आपका काम:**

इन दोनों arrays को मिलाकर (बिना सच में merge किए) उनका **median** निकालना है।

⊖ **ध्यान दें:**

Time Complexity **$O(\log(m+n))$** होनी चाहिए।

Simple merge करना allowed नहीं है।

♦ **Median क्या होता है?**

- अगर कुल elements **odd** हैं → बीच वाला element
- अगर कुल elements **even** हैं → बीच के दो elements का औसत

उदाहरण:

- `[1, 2, 3]` → median = 2
- `[1, 2, 3, 4]` → median = $(2 + 3) / 2 = 2.5$

Code

class Solution:

def findMedianSortedArrays(self, A, B):

if len(A) > len(B):

return self.findMedianSortedArrays(B, A)

m, n = len(A), len(B)

l, r = 0, m

while l <= r:

i = (l + r) // 2

j = (m + n + 1) // 2 - i

L1 = A[i-1] if i > 0 else float('-inf')

L2 = B[j-1] if j > 0 else float('-inf')

R1 = A[i] if i < m else float('inf')

R2 = B[j] if j < n else float('inf')

if L1 <= R2 and L2 <= R1:

if (m + n) % 2 == 0:

return (max(L1, L2) + min(R1, R2)) / 2

return max(L1, L2)

elif L1 > R2:

r = i - 1

else:

l = i + 1