Day – 54 of the #101 days of the coding challenge------

Problem:-

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.

Solution:-

```
double findMedianSortedArrays(vector<int>& nums1, vector<int>& nums2) {
    int m = nums1.size();
    int n = nums2.size();
    int total = m + n;
    vector<int> merged(total);

    int i = 0, j = 0, k = 0;

    while (i < m && j < n) {
        if (nums1[i] < nums2[j]) {
            merged[k++] = nums1[i++];
        } else {
            merged[k++] = nums2[j++];
        }
    }

    while (i < m) {
        merged[k++] = nums1[i++];
    }
```

```
    while (j < n) {
        merged[k++] = nums2[j++];
    }

    if (total % 2 == 0) {
        // If the total number of elements is even, return the average of the middle
two elements.
        return (merged[total / 2 - 1] + merged[total / 2]) / 2.0;
    } else {
        // If the total number of elements is odd, return the middle element.
        return merged[total / 2];
    }
}
```