**arrPos is an array that stores pairs. Each pair consists of an element from the original array and its original index.**

For example, for arr = {4, 3, 2, 1}, arrPos will initially look like {{4, 0}, {3, 1}, {2, 2}, {1, 3}}.

**Sorting**

The array arrPos is sorted based on the element values. After sorting, arrPos will become {{1, 3}, {2, 2}, {3, 1}, {4, 0}}. The first value of each pair is the element, and the second value is the original index before sorting.

**Cycle Detection**

After sorting, we want to check how the elements are rearranged by comparing their current indices with their original indices.

We use a boolean array visited[] to keep track of whether an element has already been part of a cycle.

If an element is already at the correct position (i.e., its current index matches the original index in arrPos), or if it has been visited, we skip it.

**Cycle Counting**

For each unvisited element, we follow the cycle of elements by using the original indices in arrPos. This means we move to the index of the next element until we complete a cycle.

If we find a cycle, we calculate the number of swaps needed to sort the elements in that cycle. The number of swaps for a cycle of size k is k-1.

Time Complexity: O(nlogn)

**To solve the problem of finding four elements such that a+b=c+d using sets and hash tables, we can break down the solution step by step in Java.**

You are given an array of integers. The task is to find four distinct elements

a,b,c,d such that the sum of two elements

a+b is equal to the sum of two other elements c+d.

**Approach**

**Use a HashMap to store pairs**

For every pair of elements in the array, we calculate their sum and store the pair in a HashMap. The key of the map will be the sum, and the value will be a list of pairs (indices) that result in that sum.

**Check for duplicates**

If you find that a sum already exists in the map, then you have found two pairs of elements that satisfy

a+b=c+d.

**Ensure distinct elements**

While processing pairs, ensure that the elements are distinct by checking the indices.

**Steps**

* Loop over all pairs in the array and calculate their sum.
* Use a HashMap where the sum is the key and the pair of indices is the value.
* Check if the sum already exists in the HashMap. If it does, return the result.

Time Complexity: O(n^2)