Solution 1

```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

import java.util.*;

// Average: O(n^2) time | O(n^2) space
// Worst: O(n^3) time | O(n^2) space
class Program {
  public static List<Integer[]> fourNumberSum(int[] array, int targetSum) {
    Map<Integer, List<Integer[]>> allPairSums = new HashMap<>();
    List<Integer[]> quadruplets = new ArrayList<Integer[]>();
    for (int i = 1; i < array.length - 1; i++) {
      for (int j = i + 1; j < array.length; j++) {
        int currentSum = array[i] + array[j];
        int difference = targetSum - currentSum;
        if (allPairSums.containsKey(difference)) {
          for (Integer[] pair : allPairSums.get(difference)) {
            Integer[] newQuadruplet = {pair[0], pair[1], array[i], array[j]};
            quadruplets.add(newQuadruplet);
          }
        }
      }
      for (int k = 0; k < i; k++) {
        int currentSum = array[i] + array[k];
        Integer[] pair = {array[k], array[i]};
        if (!allPairSums.containsKey(currentSum)) {
          List<Integer[]> pairGroup = new ArrayList<Integer[]>();
          pairGroup.add(pair);
          allPairSums.put(currentSum, pairGroup);
        } else {
          allPairSums.get(currentSum).add(pair);
        }
      }
    }
    return quadruplets;
  }
}

```