

Solution 1

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 // Average: O(n^2) time | O(n^2) space
4 // Worst: O(n^3) time | O(n^2) space
5 function fourNumberSum(array, targetSum) {
6   const allPairSums = {};
7   const quadruplets = [];
8   for (let i = 1; i < array.length - 1; i++) {
9     for (let j = i + 1; j < array.length; j++) {
10      const currentSum = array[i] + array[j];
11      const difference = targetSum - currentSum;
12      if (difference in allPairSums) {
13        for (const pair of allPairSums[difference]) {
14          quadruplets.push(pair.concat([array[i], array[j]]));
15        }
16      }
17    }
18    for (let k = 0; k < i; k++) {
19      const currentSum = array[i] + array[k];
20      if (!(currentSum in allPairSums)) {
21        allPairSums[currentSum] = [[array[k], array[i]]];
22      } else {
23        allPairSums[currentSum].push([array[k], array[i]]);
24      }
25    }
26  }
27  return quadruplets;
28 }
29
30 exports.fourNumberSum = fourNumberSum;
31
```