(a) Name the integers in array A $A_1, A_2, \ldots A_{n-1}, A_n$. Compute the square of integers in A and put them in another array B and name them $B_1, B_2, \ldots B_{n-1}, B_n$. Here the integers in B should be corresponding to A, which means $B_i = A_i^2$ for all integers i from 1 to n. Since there are n elements in A, we would do this step n times. $O(n)$
For each i, j from 1 to n, which i is not equal to j, calculate $A_i + B_j$, and put it in another array C. The size of C is $n(n-1)$. $O(n^2)$
We just need to compare values in C. Sort the array in time $O(n^2\, log_2\, n^2) = O(n^2 \log_2 n)$. Go through sorted C and determine if a number appear at least twice. If there is, then there exist four integers that satisfy the condition. $O(n^2 \log_2 n)$

(b) Apply the same algorithm above, except using a hash table of size $n(n-1)$ to hash elements in C and go through all of the $n(n-1)$ slots and check by brute force if the same number appears twice. $O(n^2)$