> Given an array of integers, return **indices** of the two numbers such that they add up to a specific target.

In this example, if we only want to return true if there is a solution, we can use a hash set to store all the values when we iterate the array and check if `target - current_value` is in the hash set or not.

However, we are asked to `return more information` which means we not only care about the value but also care about the index. We need to store not only the number as the key but also the index as the value. Therefore, we should use a hash map rather than a hash set.

What's More

In some cases, we need more information not just to return more information but also to `help us with our decisions`.

In the previous examples, when we meet a duplicated key, we will return the corresponding information immediately. But sometimes, we might want to check if the value of the key is acceptable first.

Template

Here we provide a template for you to solve this kind of problems:

C++ Java Copy

```
/*
 * Template for using hash map to find duplicates.
 * Replace ReturnType with the actual type of your return value.
 */
ReturnType aggregateByKey_hashmap(List<Type>& keys) {
   // Replace Type and InfoType with actual type of your key and value
   Map<Type, InfoType> hashmap = new HashMap<>();
   for (Type key : keys) {
      if (hashmap.containsKey(key)) {
         if (hashmap.get(key) satisfies the requirement) {
            return needed_information;
         }
      }
      // Value can be any information you needed (e.g. index)
      hashmap.put(key, value);
   }
   return needed_information;
}
```