Documentation

This function checks if an array of integers contains any duplicate values. It takes a list of integers as input and returns a boolean value: **True** if any value appears more than once, and **False** if all values are unique. The approach utilizes a set to track elements that have been encountered during the iteration efficiently. A set is ideal for this task because it only stores unique elements and allows for fast lookups.

The function starts by initializing an empty set, which will store each unique integer as it is encountered while iterating through the input array. For every element in the list, the function checks if the component is already present in the set. If the element is found in the set, it indicates that it has appeared before, and the function immediately returns **True**, signaling that a duplicate has been found. If the element is not in the set, it is added to the set, and the iteration continues with the next element.

If the iteration completes without finding any duplicates, meaning all elements were unique, the function returns **False**. This ensures that the function accurately identifies whether there are repeated elements in the list.

The time complexity of this approach is O(n), where **n** is the length of the input list. This is because each element is either added to the set or checked against the set, both of which are O(1) operations on average. The space complexity is also O(n) since, in the worst case, the set could store all **n** elements if no duplicates are present. This solution is efficient for large input sizes, making it a robust approach to solving the problem of finding duplicates in an integer array.