29. How do you find if ArrayList contains duplicates or not?

- There are several ways available. Shortest one is `.stream().distinct().count()` method
  `list.size() != list.stream().distinct().count()`
- Other methods:

```
//METHOD 1
public static <T> boolean containsUnique(List<T> list){ Set<T> set = new HashSet<>();
return list.stream().allMatch(t -> set.add(t));
}

//METHOD 2
public static <T> boolean containsUnique(List<T> list){ return list.stream().allMatch(new
HashSet<>()::add);
} // seems to be the best not only because it can handle pure streams, but also because it stops on
the first duplicate (while #1 and #2 always iterate till the end)

//METHOD 3
public static <T> boolean containsUnique(List<T> list){
    Set<T> set = new HashSet<>();
    for (T t: list){
        if (!set.add(t))
return false; }
```

30. Difference between Arrays and ArrayList in Java?

- Array is a part of core Java programming and has special syntax ArrayList is part of collection framework and implement List interface
- Major difference is that; Array is a fixed length data structure, so we can change length of Array one created, ArrayList is resizable.
- The other major one is that Array can contain both primitives and objects. ArrayList can only contain objects. It cannot contain primitive types.
- Also, we can compare Array and ArrayList on how to calculate length of Array or size of ArrayList. We use length for an Array, we use size() method for an ArrayList.

Array	ArrayList

int[ ] arr = {6,9,1};
• arr.length
• Arrays.sort(arry); //import java.util.Arrays
• Java also provides a convenient way to search, but only if the array is already sorted.
Arrays.binarySearch(arry, value);
• string[ ] [ ] marry = new string [3] [2];
• Arrays.asList(arry);
• Arrays.toString(arry);
• Arrays.deepToString(arry); //for multidimensional
	ArrayList list = new ArrayList();
	• list.add(obj);
	• list.add(index position, obj);
	• list.remove(obj);
	• list.set(index position, new obj); //replace object
	• list.isEmpty(); //boolean
	• list.size();
	• list.clear();
	• list.contains(obj);
	• list.get(int index);
	• list.toArray();
	• Sorting → Collection.sort(list);

31. What is thread safe or Synchronized?

- Thread safety is very important, and it is the process to make our program safe to use in multi-threaded environment, there are different ways through which we can make our program thread safe.
- **Synchronization** is the easiest and most widely used tool for thread safety.