Restriction and Rules in stream API

1. For each () 🡪 for those 11 operations, you **cannot** use for each directly because they terminate the stream and return non-stream results (like a single value, Boolean, or collection).**max**: Finds the maximum element.

1. Max :finds max element
2. **min**: Finds the minimum element.
3. **findFirst**: Finds the first element.
4. **findAny**: Finds any element (useful in parallel streams).
5. **reduce**: Combines elements to produce a single result.
6. **count**: Returns the count of elements.
7. **collect**: Gathers elements into a collection or another data structure.
8. **toArray**: Converts the stream into an array.
9. **allMatch**: Checks if all elements match a predicate.
10. **anyMatch**: Checks if any element matches a predicate.
11. **noneMatch**: Checks if no elements match a predicate.

These terminal operations end the stream pipeline and return a non-stream result.

2. Average () 🡪 if we want to count average first we need to convert into Double, or Int, Long.

 **mapToDouble()**: Converts values to double for accurate averaging.

 **mapToInt()**: Converts values to int and allows calculating an average, but it will result in an integer average.

 **mapToLong()**: Converts values to long and can also calculate the average of long values.

3. Optional 🡪

.orElse() method is used to provide a default value when an Optional is empty (i.e., no value is present). It prevents “NoSuchElementException” when accessing the value inside an Optional and ensures that the program doesn't crash.

The output of Collectors.groupingBy is always a Map.