TLE tips & Java 8

Tuesday, 4 June 2024 8:23 PM

- **List to boolean** -> Replace a List<Integer> with a boolean[] can be more efficient for certain tasks, especially when you need fast lookups and memory efficiency.
- **Use StringBuilder for String Manipulation**
- **Long instead of int**

Reminder concept in array:

- If at two different points in the array, the cumulative sums give the same remainder when divided by k, it means the sum of the elements between ↔ these two points is a multiple of k.
  - For example 🤓 lets take if we divide 7 by 5 we get remainder as 2, and if we divide 12 by 5 we also get remainder as 2, since their remainders are same (2), thus their difference which is (12 - 7 = 5) must be divisible by 5.
  - If at index i and index j (where j > i), the cumulative sums give the same remainder r when divided by k, then the sum of the subarray from i+1 to j is a multiple of k.
  - NOTE : Remember to keep reminder as positive if its negative then add k to it

Java 8

Array

- If its List<String> we can use .stream(), otherwise we have to use Arrays(arr).stream();
- For loop can also be written as IntStream
  - IntStream.range(0, value) .mapToObj(i -> String.valueOf(key)) .collect(Collectors.toList());
- Convert an array to for loop
  - array.forEach(pq::add);
- Need to used a counter?
  - Instead of int count=0, use int[] count = {0} then you can use inside Lambda
- Sort using lambda using wrapper class but need to use Arrays.fill
  - `Integer[] alpha=new Integer[26];`
  - `Arrays.fill(alpha, 0);`
  - `Arrays.sort(alpha,(a,b)->b-a);`

Map to Java 8

- map.computeIfAbsent(key, k -> new ArrayList<>()).add(value);
  - computeIfAbsent -> used for inserting and eliminates explicitly checking if key is present
  - computeIfPresent -> used for updating the value, if key is present do action otherwise ignore
- Streaming a Map, get the Keyset / valueSet / entrySet and stream it
- Getting the value inside the stream use .map or .filter and use .map(entry ->)
- The flatMap(entry -> repeatChar(entry.getKey(), entry.getValue()).stream()) calls the repeatChar method to generate a list of repeated characters for each key and then flattens these lists into a single stream.

Lambda Expression

- Map should return an integer, which will be collected as stream. So on using if conditions you can return proper integer to stream.
- .Peek method is used to see the values at each point in lambda expression -> .peek(increment -> System.out.println("increment "+increment))
- Peek is lazy its should be followed by forEach or collect otherwise it won't be executed, alternatively you can use forEach also. Or add .forEach(str -> {}) to it.
- Final or active final variable only can be used, so use Array or Object which are final