Problem 1: Group Strings by Anagrams

1. Group Strings by Anagrams using HashTable

You are organizing product codes by similarity and need to group anagrams together.

Task:
Group strings that are anagrams of each other using a HashMap.

Example:
Input:
strs = ["eat", "tea", "tan", "ate", "nat", "bat"]

Output:
[["eat", "tea", "ate"], ["tan", "nat"], ["bat"]]

Problem 1: Group Strings by Anagrams – Code

```
import java.util.*;

public class GroupAnagrams {
  public static List<List<String>> groupAnagrams(String[] strs) {
    Map<String, List<String>> map = new HashMap<>();

    for (String str : strs) {
      char[] chars = str.toCharArray();
      Arrays.sort(chars);
      String sortedStr = new String(chars);

      map.putIfAbsent(sortedStr, new ArrayList<>());
      map.get(sortedStr).add(str);
    }

    return new ArrayList<>(map.values());
  }

  public static void main(String[] args) {
    String[] strs = {"eat", "tea", "tan", "ate", "nat", "bat"};
    System.out.println(groupAnagrams(strs));
  }
}
```

Problem 1: Group Strings by Anagrams – Output

Try Run:
Input: ["eat", "tea", "tan", "ate", "nat", "bat"]
Processing:
- "eat" sorted -> "aet", add to map.
- "tea" sorted -> "aet", add to existing key.
- "tan" sorted -> "ant", add to map.
- "ate" sorted -> "aet", add to existing key.
- "nat" sorted -> "ant", add to existing key.
- "bat" sorted -> "abt", add to map.

Output: [["eat", "tea", "ate"], ["tan", "nat"], ["bat"]]

Problem 2: Longest Substring Without Repeating Characters – Question

2. Longest Substring Without Repeating Characters using Sliding Window

Given a string s, find the length of the longest substring without repeating characters.

Example 1:
Input: s = "abcabcbb"
Output: 3

Example 2:
Input: s = "bbbbb"
Output: 1

Example 3:
Input: s = "pwwkew"
Output: 3

Problem 2: Longest Substring Without Repeating Characters – Code

```
import java.util.*;

public class LongestSubstring {
   public static int lengthOfLongestSubstring(String s) {
      Set<Character> set = new HashSet<>();
      int left = 0, maxLength = 0;
```

```
        for (int right = 0; right < s.length(); right++) {
            while (set.contains(s.charAt(right))) {
                set.remove(s.charAt(left));
                left++;
            }
            set.add(s.charAt(right));
            maxLength = Math.max(maxLength, right - left + 1);
        }

        return maxLength;
    }

    public static void main(String[] args) {
        String s = "abcabcbb";
        System.out.println(lengthOfLongestSubstring(s));
    }
}
```

Problem 2: Longest Substring Without Repeating Characters – Output

Try Run:
Input: "abcabcbb"
Processing:
- "a" -> length 1
- "ab" -> length 2
- "abc" -> length 3
- "abca" -> remove "a", length 3
- Continue until max = 3

Output: 3