**Algorithms Memo (LeetCode Problem #)**

**-共通のHashMap高速探索 (49)**

|  |
| --- |
| HashMap<String, List<String>> map = new HashMap<>();  for (String s : strs) {      int[] hash = new int[26];      for (char c : s.toCharArray()) {          hash[c - 'a']++;      }      String key = new String(Arrays.toString(hash));      map.computeIfAbsent(key, k -> new ArrayList<>());      map.get(key).add(s);  } |

**-HashMapのvalueによる個数数え上げ (347)**

|  |
| --- |
| for (int num : nums) map.put(num, map.getOrDefault(num, 0) + 1); |

**-Priority Queue順位付け変更 (347)**

※Valueによる並び替え(降順)

|  |
| --- |
| PriorityQueue<Map.Entry<Integer, Integer>> pq = new PriorityQueue<>((a, b) -> b.getValue() - a.getValue()); |

※元々Priority Queueは昇順での並びで先頭が決まるのでb – aとし、降順を実装した。

**-HashSetによる高速数列探し (128)**

|  |
| --- |
| HashSet<Integer> set = new HashSet<>();  int ans = 1;  for (int num : nums) set.add(num);  for (int num : nums) {      if (!set.contains(num - 1)) {          int count = 1;          while (set.contains(num + 1)) {              num++;              count++;          }          ans = Math.max(count, ans);      }  } |

if (!set.contains(num - 1))

により、数列の最初の項のみの処理にできる

※双方向で考える必要がない

-