,

目录

第 1 章
Graph

1.1 Alien Dictionary

Description

There is a new alien language which uses the latin alphabet. However, the order among letters are unknown to you. You receive a list of non-empty words from the dictionary, where words are sorted lexicographically by the rules of this new language. Derive the order of letters in this language.

Example:

Given the following words in dictionary,

```
[
  "wrt",
  "wrf",
  "er",
  "ett",
  "rftt"
]
```

The correct order is: `"wertf"`.

Note:

1. You may assume all letters are in lowercase.
2. You may assume that if a is a prefix of b, then a must appear before b in the given dictionary.
3. If the order is invalid, return an empty string.
4. There may be multiple valid order of letters, return any one of them is fine.

Solution

```
public String alienOrder(String[] words) {
    int[] indegree = new int[26];
    Arrays.fill(indegree, -1);

    int count = 0;
    for (String word : words) {
        for (char c : word.toCharArray()) {
            if (indegree[c - 'a'] != 0) {
                indegree[c - 'a'] = 0;
                count++;
            }
        }
    }
    HashMap<Character, Set<Character>> map = new HashMap<>();
    for (int i = 0; i < words.length - 1; i++) {
        String first = words[i], second = words[i + 1];
        int len = Math.min(first.length(), second.length());
        for (int j = 0; j < len; j++) {
            if (first.charAt(j) != second.charAt(j)) {
                Set<Character> set = map.get(first.charAt(j));
                if (set == null) {
                    set = new HashSet<Character>();
                    map.put(first.charAt(j), set);
                }
                if (set.add(second.charAt(j))) {
                    indegree[second.charAt(j) - 'a']++;
                }
                break;
            } else {
                if (j + 1 >= second.length() && j + 1 < first.length()) { return ""; }
            }
        }
    }
    Queue<Character> queue = new LinkedList<Character>();
    for (int i = 0; i < indegree.length; i++) {
        if (indegree[i] == 0) {
            queue.add((char) ('a' + i));
        }
    }
    StringBuilder sb = new StringBuilder();
    while (!queue.isEmpty()) {
        Character from = queue.poll();
        sb.append(from);
        Set<Character> set = map.get(from);
        if (set != null) {
            for (Character to : map.get(from)) {
                if (--indegree[to - 'a'] == 0) {
                    queue.add(to);
                }
            }
        }
    }
    return sb.length() != count ? "" : sb.toString();
}
```