

Java-数组排序字符串-单词接龙

题目描述：单词接龙的规则是：可用于接龙的单词首字母必须要前一个单词的尾字母相同；当存在多个首字母相同的单词时，取长度最长的单词，如果长度也相等，则取字典序最小的单词；已经参与接龙的单词不能重复使用。现给定一组全部由小写字母组成单词数组，并指定其中的一个单词作为起始单词，进行单词接龙，请输出最长的单词串，单词串是单词拼接而成，中间没有空格。

输入描述：输入的第一行为一个非负整数，表示起始单词在数组中的索引 K ， $0 \leq K < N$ ；输入的第二行为一个非负整数，表示单词的个数 N ；接下来的 N 行，分别表示单词数组中的单词。

输出描述：输出一个字符串，表示最终拼接的单词串。

补充说明：单词个数 N 的取值范围为 $[1, 20]$ ；单个单词的长度的取值范围为 $[1, 30]$ ；

示例

展开

示例1

输入：0

6

word

dd

da

dc

dword

d

输出：worddwordda

说明：先确定起始单词word，再接以d开头的且长度最长的单词dword，剩余以d开头且长度最长的有dd、da、dc，则取字典序最小的da，所以最后输出worddwordda。

示例2

输入：4

6

word

dd

da

dc

dword

d

输出：dwordda

说明：先确定起始单词dword，剩余以d开头且长度最长的有dd、da、dc，则取字典序最小的da，所以最后输出dwordda。

```

1  import java.util.*;
2
3  public class Main {
4      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6          int start = scanner.nextInt();
7          scanner.nextLine();
8          int N = scanner.nextInt();
9          scanner.nextLine();
10         HashMap<Character, LinkedList<String>> map = new HashMap<>();
11         StringBuilder sb = new StringBuilder();
12         for (int i = 0; i < N; i++) {
13             String s = scanner.nextLine();
14             if (i==start){
15                 sb.append(s);
16             }else {
17                 char c = s.charAt(0);
18                 LinkedList<String> strings = map.get(c);
19                 if (strings!=null){
20                     strings.add(s);
21                 }else {
22                     LinkedList<String> list = new LinkedList<>();
23                     list.add(s);
24                     map.put(c,list);
25                 }
26                 // List<String> list = map.getDefault(s.charAt(0), new LinkedList<>());
27                 // list.add(s);
28             }
29         }
30         for (LinkedList<String> value : map.values()) {
31             // System.out.println(value);
32             value.sort((a,b)->{
33                 if (a.length()==b.length()){
34                     return a.compareTo(b);
35                 }
36                 return b.length()-a.length();
37             });
38         }
39         // for (Map.Entry<Character, LinkedList<String>> entry : map.entrySet()) {
40         //     System.out.println(entry.getKey()+" "+entry.getValue());
41         // }
42         while (true){
43             char c = sb.charAt(sb.length()-1);
44             LinkedList<String> strings = map.get(c);
45             if (strings!=null&&strings.size()>0){
46                 sb.append(strings.getFirst());
47                 strings.removeFirst();
48             }else {
49                 break;
50             }
51         }
52         System.out.println(sb);
53     }
54 }

```