```
1  // Computes a map in which the keys are words and values are Lists of words
2  // that differ in only one character from the corresponding key.
3  // Uses an efficient algorithm that is O(N log N) with a TreeMap.
4  public static Map<String,List<String>>
5  computeAdjacentWords( List<String> words )
6  {
7      Map<String,List<String>> adjWords = new TreeMap<String,List<String>>( );
8      Map<Integer,List<String>> wordsByLength =
9                                          new TreeMap<Integer,List<String>>( );
10
11       // Group the words by their length
12     for( String w : words )
13         update( wordsByLength, w.length( ), w );
14
15       // Work on each group separately
16     for( Map.Entry<Integer,List<String>> entry : wordsByLength.entrySet( ) )
17     {
18         List<String> groupsWords = entry.getValue( );
19         int groupNum = entry.getKey( );
20
21         // Work on each position in each group
22         for( int i = 0; i < groupNum; i++ )
23         {
24             // Remove one character in specified position, computing
25             // representative.  Words with same representative are
26             // adjacent, so first populate a map ...
27             Map<String,List<String>> repToWord =
28                                        new TreeMap<String,List<String>>( );
29
30             for( String str : groupsWords )
31             {
32                 String rep = str.substring( 0, i ) + str.substring( i + 1 );
33                 update( repToWord, rep, str );
34             }
35
36             // and then look for map values with more than one string
37             for( List<String> wordClique : repToWord.values( ) )
38                 if( wordClique.size( ) >= 2 )
39                     for( String s1 : wordClique )
40                         for( String s2 : wordClique )
41                             if( s1 != s2 )
42                                 update( adjWords, s1, s2 );
43         }
44     }
45
46     return adjWords;
47 }
```

图 4-68　计算包含单词作为关键字及只有一个字母不同的一列单词作为值的映射的函数。对一个 89 000 单词的词典只运行 4 秒钟