ImplementTrie.java

```
1   package Trie;
2
3   class Node {
4       Node links[] = new Node[26];
5       int cntEndWith = 0;
6       int cntPrefix = 0;
7
8       public Node() {
9       }
10
11      boolean containsKey(char ch) {
12  3       return (links[ch - 'a'] != null);
13      }
14      Node get(char ch) {
15  2       return links[ch-'a'];
16      }
17       void put(char ch, Node node) {
18  1       links[ch-'a'] = node;
19
20      }
21       void increaseEnd() {
22  1       cntEndWith++;
23      }
24       void increasePrefix() {
25  1       cntPrefix++;
26      }
27       void deleteEnd() {
28  1       cntEndWith--;
29      }
30       void reducePrefix() {
31  1       cntPrefix--;
32      }
33       int getEnd() {
34  1       return cntEndWith;
35      }
36       int getPrefix() {
37  1       return cntPrefix;
38      }
39  };
40  public class ImplementTrie {
41
42
43
44              private Node root;
45
46              //Initialize your data structure here
47
48              ImplementTrie() {
49                  root = new Node();
50              }
51
52
53              //Inserts a word into the trie
```

```
54
55              public void insert(String word) {
56                  Node node = root;
57 2                for(int i = 0;i<word.length();i++) {
58 1                  if(!node.containsKey(word.charAt(i))) {
59 1                      node.put(word.charAt(i), new Node());
60                      }
61                      node = node.get(word.charAt(i));
62 1                  node.increasePrefix();
63                  }
64 1                node.increaseEnd();
65              }
66
67
68              public int countWordsEqualTo(String word) {
69                  Node node = root;
70 2                for(int i = 0;i<word.length();i++) {
71 1                  if(node.containsKey(word.charAt(i))) {
72                      node = node.get(word.charAt(i));
73                      }
74                      else {
75                          return 0;
76                      }
77                  }
78 1                return node.getEnd();
79              }
80
81              public int countWordsStartingWith(String word) {
82                  Node node = root;
83 2                for(int i = 0;i<word.length();i++) {
84 1                  if(node.containsKey(word.charAt(i))) {
85                      node = node.get(word.charAt(i));
86                      }
87                      else {
88                          return 0;
89                      }
90                  }
91 1                return node.getPrefix();
92              }
93
94              public void erase(String word) {
95                  Node node = root;
96 2                for(int i = 0;i<word.length();i++) {
97 1                  if(node.containsKey(word.charAt(i))) {
98                      node = node.get(word.charAt(i));
99 1                      node.reducePrefix();
100                     }
101                     else {
102                         return;
103                     }
104                 }
105 1                node.deleteEnd();
106             }
107 }
```

Mutations

12	1. negated conditional → KILLED 2. Replaced integer subtraction with addition → KILLED 3. replaced boolean return with true for Trie/Node::containsKey → KILLED
15	1. Replaced integer subtraction with addition → KILLED 2. replaced return value with null for Trie/Node::get → KILLED
18	1. Replaced integer subtraction with addition → KILLED
22	1. Replaced integer addition with subtraction → KILLED
25	1. Replaced integer addition with subtraction → KILLED
28	1. Replaced integer subtraction with addition → KILLED
31	1. Replaced integer subtraction with addition → SURVIVED
34	1. replaced int return with 0 for Trie/Node::getEnd → KILLED
37	1. replaced int return with 0 for Trie/Node::getPrefix → KILLED
57	1. negated conditional → KILLED 2. changed conditional boundary → KILLED
58	1. negated conditional → KILLED
59	1. removed call to Trie/Node::put → KILLED
62	1. removed call to Trie/Node::increasePrefix → KILLED
64	1. removed call to Trie/Node::increaseEnd → KILLED
70	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
71	1. negated conditional → KILLED
78	1. replaced int return with 0 for Trie/ImplementTrie::countWordsEqualTo → KILLED
83	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
84	1. negated conditional → KILLED
91	1. replaced int return with 0 for Trie/ImplementTrie::countWordsStartingWith → KILLED
96	1. negated conditional → KILLED 2. changed conditional boundary → KILLED
97	1. negated conditional → KILLED
99	1. removed call to Trie/Node::reducePrefix → SURVIVED
105	1. removed call to Trie/Node::deleteEnd → KILLED

Active mutators

- CONDITIONALS_BOUNDARY
- EMPTY_RETURNS
- FALSE_RETURNS
- INCREMENTS
- INVERT_NEGS
- MATH
- NEGATE_CONDITIONALS
- NULL_RETURNS
- PRIMITIVE_RETURNS
- TRUE_RETURNS
- VOID_METHOD_CALLS

Tests examined

- Trie.ImplementTrieTest.implementTrieTest(Trie.ImplementTrieTest) (0 ms)
- Trie.NumberofDistinctSubstringTest.main(Trie.NumberofDistinctSubstringTest) (0 ms)

Report generated by PIT 1.15.0