MaximumXorOfTwo.java

```
1   package com.example.trie;
2
3   import java.util.ArrayList;
4
5   class Node1 {
6       Node1 links[] = new Node1[2];
7
8       public Node1() {
9       }
10
11      boolean containsKey(int ind) {
12 2        return (links[ind] != null);
13      }
14
15      Node1 get(int ind) {
16 1        return links[ind];
17      }
18
19      void put(int ind, Node1 node) {
20          links[ind] = node;
21      }
22  };
23
24  class Trie {
25      private static Node1 root;
26
27      // Initialize your data structure here
28      Trie() {
29          root = new Node1();
30      }
31
32      // Inserts a word into the trie
33      public void insert(int num) {
34          Node1 node = root;
35 2        for (int i = 31; i >= 0; i--) {
36 2            int bit = (num >> i) & 1;
37 1            if (!node.containsKey(bit)) {
38 1                node.put(bit, new Node1());
39              }
40              node = node.get(bit);
41          }
42      }
43
44      public int getMax(int num) {
45          Node1 node = root;
46          int maxNum = 0;
47 2        for (int i = 31; i >= 0; i--) {
48 2            int bit = (num >> i) & 1;
49 2            if (node.containsKey(1 - bit)) {
50 2                maxNum = maxNum | (1 << i);
51 1                node = node.get(1 - bit);
52              } else {
53                  node = node.get(bit);
54              }
55          }
56 1        return maxNum;
57      }
58
59  };
60
61  public class MaximumXorOfTwo {
62
```

```
63      public int maxXOR(int n, int m, ArrayList<Integer> arr1, ArrayList<Integer> arr2) {
64          Trie trie = new Trie();
65 2        for (int i = 0; i < n; i++) {
66 1            trie.insert(arr1.get(i));
67          }
68          int maxi = 0;
69 2        for (int i = 0; i < m; i++) {
70              maxi = Math.max(maxi, trie.getMax(arr2.get(i)));
71          }
72 1        return maxi;
73      }
74  }
```

Mutations

12	1. replaced boolean return with true for com/example/trie/Node1::containsKey → KILLED 2. negated conditional → KILLED
16	1. replaced return value with null for com/example/trie/Node1::get → KILLED
35	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
36	1. Replaced bitwise AND with OR → KILLED 2. Replaced Shift Right with Shift Left → KILLED
37	1. negated conditional → KILLED
38	1. removed call to com/example/trie/Node1::put → KILLED
47	1. negated conditional → KILLED 2. changed conditional boundary → KILLED
48	1. Replaced Shift Right with Shift Left → KILLED 2. Replaced bitwise AND with OR → KILLED
49	1. Replaced integer subtraction with addition → KILLED 2. negated conditional → KILLED
50	1. Replaced Shift Left with Shift Right → KILLED 2. Replaced bitwise OR with AND → KILLED
51	1. Replaced integer subtraction with addition → KILLED
56	1. replaced int return with 0 for com/example/trie/Trie::getMax → KILLED
65	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
66	1. removed call to com/example/trie/Trie::insert → KILLED
69	1. negated conditional → KILLED 2. changed conditional boundary → KILLED
72	1. replaced int return with 0 for com/example/trie/MaximumXorOfTwo::maxXOR → 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

- com.example.trie.MaximumXorOfTwoTest.testtrie(com.example.trie.MaximumXorOfTwoTest) (0 ms)

Report generated by PIT 1.15.0