Graph.java

```
1   package com.hongchuan.app;
2   // Java program to print DFS traversal from a given given graph
3   import java.io.*;
4   import java.util.*;
5
6   // This class represents a directed graph using adjacency list
7   // representation
8   public class Graph
9   {
10      private int V;   // No. of vertices
11
12      // Array  of lists for Adjacency List Representation
13      private LinkedList<Integer> adj[];
14
15      ArrayList <Integer> trace = new ArrayList<Integer>(0);
16      // Constructor
17      public Graph(int v)
18      {
19          V = v;
20          adj = new LinkedList[v];
21 3        for (int i=0; i<v; ++i)
22              adj[i] = new LinkedList();
23      }
24
25      public ArrayList<Integer> getTrace(){
26 1          return trace;
27      }
28      //Function to add an edge into the graph
29      public void addEdge(int v, int w)
30      {
31          adj[v].add(w);  // Add w to v's list.
32      }
33
34      // A function used by DFS
35      public void DFSUtil(int v,boolean visited[])
36      {
37          // Mark the current node as visited and print it
38          visited[v] = true;
39          trace.add(v);
40          // System.out.print(v+" ");
41
42          // Recur for all the vertices adjacent to this vertex
43          Iterator<Integer> i = adj[v].listIterator();
44 1        while (i.hasNext())
```

```
45          {
46              int n = i.next();
47 1            if (!visited[n])
48 1                DFSUtil(n, visited);
49          }
50      }
51
52      // The function to do DFS traversal. It uses recursive DFSUtil()
53      public void DFS(int v)
54      {
55          // Mark all the vertices as not visited(set as
56          // false by default in java)
57          boolean visited[] = new boolean[V];
58
59          // Call the recursive helper function to print DFS traversal
60 1        DFSUtil(v, visited);
61      }
62
63      // public static void main(String args[])
64      // {
65      //     Graph g = new Graph(4);
66
67      //     g.addEdge(0, 1);
68      //     g.addEdge(0, 2);
69      //     g.addEdge(1, 2);
70      //     g.addEdge(2, 0);
71      //     g.addEdge(2, 3);
72      //     g.addEdge(3, 3);
73
74      //     System.out.println("Following is Depth First Traversal "+
75      //                        "(starting from vertex 2)");
76
77      //     g.DFS(0);
78      //     System.out.println(g.getTrace());
79      // }
80  }
81  // This code is contributed by Aakash Hasija
```

Mutations

1. changed conditional boundary → KILLED

21 2. Changed increment from 1 to -1 → KILLED

3. negated conditional → KILLED

1. mutated return of Object value for

26 com/hongchuan/app/Graph::getTrace to (if (x != null) null else throw new RuntimeException) → KILLED

44 1. negated conditional → SURVIVED

47 1. negated conditional → SURVIVED

48 1. removed call to com/hongchuan/app/Graph::DFSUtil → SURVIVED

```
60  1. removed call to com/hongchuan/app/Graph::DFSUtil → KILLED
```

Active mutators

- INCREMENTS_MUTATOR
- VOID_METHOD_CALL_MUTATOR
- RETURN_VALS_MUTATOR
- MATH_MUTATOR
- NEGATE_CONDITIONALS_MUTATOR
- INVERT_NEGS_MUTATOR
- CONDITIONALS_BOUNDARY_MUTATOR

Tests examined

- com.hongchuan.app.GraphTest.testCase3(com.hongchuan.app.GraphTest) (7 ms)

Report generated by PIT 1.4.3