Paths.java

```
   package com.example;

   import java.util.*;

   class Paths {
       public int MOD = (int)(1e9 +7);
       public int countPaths1(int n, int[][] roads) {
           ArrayList<ArrayList<Pair>> list = new ArrayList<>();
3          for(int i=0;i<n;i++){
               ArrayList<Pair> ap=new ArrayList<Pair>();
               list.add(ap);
           }
2          for(int i=0;i<roads.length;i++){
               list.get(roads[i][0]).add(new Pair(roads[i][1],(long) roads[i][2]));
               list.get(roads[i][1]).add(new Pair(roads[i][0], (long)roads[i][2]));
           }
           PriorityQueue<Pair> pq= new PriorityQueue<>();
           pq.add(new Pair(0,0));
           long[] dist= new long[n];
1          Arrays.fill(dist, Long.MAX_VALUE/2);
           dist[0]=0;
           int[] ways = new int[n];
           ways[0]=1;
          // int minD=Integer.MAX_VALUE;
1          while(!pq.isEmpty()){
               Pair p = pq.peek();
               int node =p.node;
               long d=p.d;
               pq.poll();
               for(Pair np: list.get(node)){
                   int newNode = np.node;
                   long newD =np.d;
3                  if(d+newD<dist[newNode]){
                       ways[newNode]= ways[node];
1                      dist[newNode]=d+newD;
1                      pq.add(new Pair(newNode,d+newD));
                   }
2                  else if(d+newD==dist[newNode]){
2                     ways[newNode]=(ways[newNode]  + ways[node])%MOD;
                   }
               }
           }
3          return ways[n-1]%MOD;
       }
   }

   class Pair implements Comparable<Pair> {
       int node;
       long d;

       Pair(int node, long d) {
           this.node = node;
           this.d = d;
       }

       @Override
       public int compareTo(Pair other) {
```

```
58 1         return Long.compare(this.d, other.d);
59       }
60   }
```

Mutations

```
     1. changed conditional boundary → SURVIVED
9    2. Changed increment from 1 to -1 → TIMED_OUT
     3. negated conditional → KILLED
13   1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
20   1. removed call to java/util/Arrays::fill → KILLED
25   1. negated conditional → KILLED
     1. changed conditional boundary → SURVIVED
33   2. Replaced long addition with subtraction → TIMED_OUT
     3. negated conditional → KILLED
35   1. Replaced long addition with subtraction → SURVIVED
36   1. Replaced long addition with subtraction → TIMED_OUT
38   1. Replaced long addition with subtraction → KILLED
     2. negated conditional → KILLED
39   1. Replaced integer addition with subtraction → SURVIVED
     2. Replaced integer modulus with multiplication → SURVIVED
     1. Replaced integer subtraction with addition → KILLED
43   2. Replaced integer modulus with multiplication → KILLED
     3. replaced int return with 0 for com/example/Paths::countPaths1 → KILLED
58   1. replaced int return with 0 for com/example/Pair::compareTo → SURVIVED
```

Active mutators

- BOOLEAN_FALSE_RETURN
- BOOLEAN_TRUE_RETURN
- CONDITIONALS_BOUNDARY_MUTATOR
- EMPTY_RETURN_VALUES
- INCREMENTS_MUTATOR
- INVERT_NEGS_MUTATOR
- MATH_MUTATOR
- NEGATE_CONDITIONALS_MUTATOR
- NULL_RETURN_VALUES
- PRIMITIVE_RETURN_VALS_MUTATOR
- VOID_METHOD_CALL_MUTATOR

Tests examined

- com.example.PathsTest.testCountPaths6(com.example.PathsTest) (1 ms)
- com.example.PathsTest.testCountPaths4(com.example.PathsTest) (0 ms)
- com.example.PathsTest.testCountPaths7(com.example.PathsTest) (0 ms)
- com.example.PathsTest.testCountPaths8(com.example.PathsTest) (0 ms)
- com.example.PathsTest.testCountPaths9(com.example.PathsTest) (0 ms)

Report generated by PIT 1.5.0