LIS.java

```
1   package DynamicProgramming;
2
3   import java.util.Arrays;
4
5   public class LIS {
6
7       public static int findLISLen(int a[]) {
8           int size = a.length;
9           int arr[] = new int[size];
10          arr[0] = a[0];
11          int lis = 1;
12 2        for (int i = 1; i < size; i++) {
13              int index = binarySearchBetween(arr, lis, a[i]);
14              arr[index] = a[i];
15 2            if (index > lis) {
16 1                lis++;
17              }
18          }
19 1        return lis;
20      }
21
22      // O(logn)
23      public static int binarySearchBetween(int[] t, int end, int key) {
24          int left = 0;
25          int right = end;
26 2        if (key < t[0]) {
27              return 0;
28          }
29 2        if (key > t[end]) {
30 2            return end + 1;
31          }
32 3        while (left < right - 1) {
33 2            int middle = (left + right) / 2;
34 2            if (t[middle] < key) {
35                  left = middle;
36              } else {
37                  right = middle;
38              }
39          }
40 1        return right;
41      }
42
43      public static int dp(int[] nums) {
44          int[] dp = new int[nums.length];
45 1        Arrays.fill(dp, 1);
46
47 2        for (int i = 1; i < nums.length; i++) {
48 2            for (int j = 0; j < i; j++) {
49 2                if (nums[i] > nums[j]) {
50 1                    dp[i] = Math.max(dp[i], dp[j] + 1);
```

```
51                  }
52              }
53          }
54
55          int longest = 0;
56          for (int c: dp) {
57              longest = Math.max(longest, c);
58          }
59
60 1        return longest;
61      }
62
63      public static int rec(int[] nums, int n, int curr, int prev){
64 2        if(curr >= n) return 0;
65          int left = 0;
66 3        if(prev == Integer.MIN_VALUE || prev <nums[curr])
67 2            left = rec(nums, n, curr+1, nums[curr])+1;
68 1        int right = rec(nums, n, curr+1, prev);
69          int val = Math.max(left, right);
70 1        return val;
71      }
72  }
```

Mutations

12	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
15	1. negated conditional → KILLED 2. changed conditional boundary → KILLED
16	1. Changed increment from 1 to -1 → KILLED
19	1. replaced int return with 0 for DynamicProgramming/LIS::findLISLen → KILLED
26	1. negated conditional → KILLED 2. changed conditional boundary → SURVIVED
29	1. changed conditional boundary → SURVIVED 2. negated conditional → KILLED
30	1. replaced int return with 0 for DynamicProgramming/LIS::binarySearchBetween → KILLED 2. Replaced integer addition with subtraction → KILLED
32	1. Replaced integer subtraction with addition → TIMED_OUT 2. changed conditional boundary → TIMED_OUT 3. negated conditional → SURVIVED
33	1. Replaced integer addition with subtraction → KILLED 2. Replaced integer division with multiplication → KILLED
34	1. negated conditional → SURVIVED 2. changed conditional boundary → SURVIVED
40	1. replaced int return with 0 for DynamicProgramming/LIS::binarySearchBetween → SURVIVED
45	1. removed call to java/util/Arrays::fill → KILLED
47	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
48	1. changed conditional boundary → SURVIVED 2. negated conditional → KILLED
49	1. changed conditional boundary → SURVIVED 2. negated conditional → KILLED
50	1. Replaced integer addition with subtraction → KILLED
60	1. replaced int return with 0 for DynamicProgramming/LIS::dp → KILLED

64	1. negated conditional → KILLED 2. changed conditional boundary → KILLED
66	1. changed conditional boundary → SURVIVED 2. negated conditional → KILLED 3. negated conditional → KILLED
67	1. Replaced integer addition with subtraction → KILLED 2. Replaced integer addition with subtraction → KILLED
68	1. Replaced integer addition with subtraction → KILLED
70	1. replaced int return with 0 for DynamicProgramming/LIS::rec → 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

- DynamicProgramming.LISTest.testBinarySearch(DynamicProgramming.LISTest) (0 ms)
- DynamicProgramming.LISTest.testDP(DynamicProgramming.LISTest) (0 ms)
- DynamicProgramming.LISTest.testRec(DynamicProgramming.LISTest) (0 ms)

Report generated by PIT 1.15.0