HouseRobber.java

```
  package DynamicProgramming;

  import java.util.*;
  //Memoization
  class Solution1 {

      public int money(int i, int nums[], int[]dp){
2         if(i>=nums.length) return 0;

4         if(i == nums.length-1) return nums[nums.length-1];
2         if(dp[i] != -1) return dp[i];
4         return dp[i] = Math.max( (nums[i] + money(i+2, nums, dp)), (money(i+1, nums, dp)) );

      }

      public int rob(int[] nums) {
1         int dp1[] = new int[nums.length+1];
1         int dp2[] = new int[nums.length+1];
1         Arrays.fill(dp1, -1);
1         Arrays.fill(dp2, -1);
2         if(nums.length==1) return nums[0];
1         int [] nums1 = new int[nums.length-1];
1         int [] nums2 = new int[nums.length-1];
2         for(int i=1; i<nums.length; i++){
1              nums1[i-1] = nums[i];
          }
3         for(int i=0; i<nums.length-1; i++){
               nums2[i] = nums[i];
          }
1         return Math.max( money(0,nums1,dp1), money(0, nums2, dp2));
      }

  }

  //Tabulation

  class Solution2 {
      public int rob(int[] nums) {
          int n = nums.length;
          // if(n==0) return 0;
2         if(n==1) return nums[0];
          // if(n==2) return Math.max(nums[0], nums[1]);
1         int [] nums1 = new int[nums.length-1];
1         int [] nums2 = new int[nums.length-1];
2         for(int i=1; i<nums.length; i++){
1              nums1[i-1] = nums[i];
          }
3         for(int i=0; i<nums.length-1; i++){
               nums2[i] = nums[i];
          }
          int dp1[] = new int[nums.length];
          int dp2[] = new int[nums.length];
1         Arrays.fill(dp1, -1);
1         Arrays.fill(dp2, -1);
          dp1[nums1.length] = 0;
2         dp1[nums1.length-1] = nums1[nums1.length-1];
          dp2[nums2.length] = 0;
2         dp2[nums2.length-1] = nums2[nums2.length-1];

1         return Math.max( money(nums1, dp1), money(nums2, dp2) );

      }
      public int money(int nums[], int[]dp){
3         for(int i=nums.length-2; i>=0; i--){
3             dp[i] = Math.max((nums[i] + dp[i+2]), dp[i+1]);
          }
67
68  1         return dp[0];
69        }
70    }
71
72    //Tabulation with optimization
73    class Solution3 {
74        public int rob(int[] nums) {
75            int n = nums.length;
76  2         if(n==1) return nums[0];
77
78  1         int [] nums1 = new int[nums.length-1];
79  1         int [] nums2 = new int[nums.length-1];
80  2         for(int i=1; i<nums.length; i++){
81  1              nums1[i-1] = nums[i];
82            }
83  3         for(int i=0; i<nums.length-1; i++){
84                 nums2[i] = nums[i];
85            }
86
87  1         return Math.max( money(nums1), money(nums2) );
88
89        }
90        public int money(int nums[]){
91            int prev =0;
92  1         int next = nums[nums.length-1];
93            int curr;
94  3         for(int i=nums.length-2; i>=0; i--){
95  1             curr = Math.max((nums[i] + prev), next);
96                prev = next;
97                next = curr;
98            }
99
100 1         return next;
101       }
102   }
```

Mutations

```
8   1. negated conditional → KILLED
    2. changed conditional boundary → KILLED
10  1. replaced int return with 0 for DynamicProgramming/Solution1::money → KILLED
    2. Replaced integer subtraction with addition → KILLED
    3. Replaced integer subtraction with addition → SURVIVED
    4. negated conditional → KILLED
11  1. replaced int return with 0 for DynamicProgramming/Solution1::money → NO_COVERAGE
    2. negated conditional → KILLED
12  1. Replaced integer addition with subtraction → KILLED
    2. Replaced integer addition with subtraction → KILLED
    3. Replaced integer addition with subtraction → KILLED
    4. replaced int return with 0 for DynamicProgramming/Solution1::money → KILLED
17  1. Replaced integer addition with subtraction → SURVIVED
18  1. Replaced integer addition with subtraction → SURVIVED
19  1. removed call to java/util/Arrays::fill → SURVIVED
20  1. removed call to java/util/Arrays::fill → KILLED
21  1. replaced int return with 0 for DynamicProgramming/Solution1::rob → SURVIVED
    2. negated conditional → KILLED
22  1. Replaced integer subtraction with addition → SURVIVED
23  1. Replaced integer subtraction with addition → SURVIVED
24  1. negated conditional → KILLED
    2. changed conditional boundary → KILLED
25  1. Replaced integer subtraction with addition → KILLED
27  1. changed conditional boundary → KILLED
    2. Replaced integer subtraction with addition → KILLED
    3. negated conditional → KILLED
30  1. replaced int return with 0 for DynamicProgramming/Solution1::rob → KILLED
41  1. negated conditional → KILLED
    2. replaced int return with 0 for DynamicProgramming/Solution2::rob → SURVIVED
43  1. Replaced integer subtraction with addition → KILLED
44  1. Replaced integer subtraction with addition → KILLED
45  1. changed conditional boundary → KILLED
    2. negated conditional → KILLED
46  1. Replaced integer subtraction with addition → KILLED
```

48	1. negated conditional → KILLED 2. changed conditional boundary → KILLED 3. Replaced integer subtraction with addition → KILLED
53	1. removed call to java/util/Arrays::fill → SURVIVED
54	1. removed call to java/util/Arrays::fill → SURVIVED
56	1. Replaced integer subtraction with addition → KILLED 2. Replaced integer subtraction with addition → KILLED
58	1. Replaced integer subtraction with addition → KILLED 2. Replaced integer subtraction with addition → KILLED
60	1. replaced int return with 0 for DynamicProgramming/Solution2::rob → KILLED
64	1. changed conditional boundary → KILLED 2. negated conditional → KILLED 3. Replaced integer subtraction with addition → KILLED
65	1. Replaced integer addition with subtraction → KILLED 2. Replaced integer addition with subtraction → KILLED 3. Replaced integer addition with subtraction → KILLED
68	1. replaced int return with 0 for DynamicProgramming/Solution2::money → KILLED
76	1. negated conditional → KILLED 2. replaced int return with 0 for DynamicProgramming/Solution3::rob → SURVIVED
78	1. Replaced integer subtraction with addition → SURVIVED
79	1. Replaced integer subtraction with addition → SURVIVED
80	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
81	1. Replaced integer subtraction with addition → KILLED
83	1. negated conditional → KILLED 2. Replaced integer subtraction with addition → KILLED 3. changed conditional boundary → KILLED
87	1. replaced int return with 0 for DynamicProgramming/Solution3::rob → KILLED
92	1. Replaced integer subtraction with addition → KILLED
94	1. Replaced integer subtraction with addition → KILLED 2. negated conditional → KILLED 3. changed conditional boundary → KILLED
95	1. Replaced integer addition with subtraction → KILLED
100	1. replaced int return with 0 for DynamicProgramming/Solution3::money → 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.HouseRobberTest.testSolution3(DynamicProgramming.HouseRobberTest) (0 ms)
- DynamicProgramming.HouseRobberTest.testSolution1(DynamicProgramming.HouseRobberTest) (0 ms)
- DynamicProgramming.HouseRobberTest.testSolution2(DynamicProgramming.HouseRobberTest) (0 ms)

Report generated by PIT 1.15.0