

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1	Solution 2	Solution 3
<pre>1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved. 2 3 #include <algorithm> 4 #include <vector> 5 #include <numeric> 6 using namespace std; 7 8 vector<int> getLocalMinIdxs(vector<int> array); 9 void expandFromLocalMinIdx(int localMinIdx, vector<int> scores, 10 vector<int> *rewards); 11 12 // O(n) time O(n) space - where in is the length of the input array 13 int minRewards(vector<int> scores) { 14 vector<int> rewards = vector<int>(scores.size(), 1); 15 vector<int> localMinIdxs = getLocalMinIdxs(scores); 16 for (int localMinIdx : localMinIdxs) { 17 expandFromLocalMinIdx(localMinIdx, scores, &rewards); 18 } 19 return accumulate(rewards.begin(), rewards.end(), 0); 20 } 21 22 vector<int> getLocalMinIdxs(vector<int> array) { 23 if (array.size() == 1) 24 return vector<int>{0}; 25 vector<int> localMinIdxs = {}; 26 for (int i = 0; i < array.size(); i++) { 27 if (i == 0 && array[i] < array[i + 1]) 28 localMinIdxs.push_back(i); 29 if (i == array.size() - 1 && array[i] < array[i - 1]) 30 localMinIdxs.push_back(i); 31 if (i == 0 i == array.size() - 1) 32 continue; 33 if (array[i] < array[i + 1] && array[i] < array[i - 1])</pre>		<pre>1 #include <vector> 2 using namespace std; 3 4 int minRewards(vector<int> scores) { 5 // Write your code here. 6 return -1; 7 } 8</pre>

Our Tests

Custom Output

Submit Code

1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.

2

3 #include <algorithm>

4 #include <vector>

5 #include <numeric>

6 using namespace std;

7

8 vector<int> getLocalMinIdxs(vector<int> array);

9 void expandFromLocalMinIdx(int localMinIdx, vector<int> scores,

10 vector<int> *rewards);

11

12 // O(n) time | O(n) space - where in is the length of the input array

13 int minRewards(vector<int> scores) {

14 vector<int> rewards = vector<int>(scores.size(), 1);

15 vector<int> localMinIdxs = getLocalMinIdxs(scores);

16 for (int localMinIdx : localMinIdxs) {

17 expandFromLocalMinIdx(localMinIdx, scores, &rewards);

18 }

19 return accumulate(rewards.begin(), rewards.end(), 0);

20 }

21

22 vector<int> getLocalMinIdxs(vector<int> array) {

23 if (array.size() == 1)

24 return vector<int>{0};

25 vector<int> localMinIdxs = {};

26 for (int i = 0; i < array.size(); i++) {

27 if (i == 0 && array[i] < array[i + 1])

28 localMinIdxs.push_back(i);

29 if (i == array.size() - 1 && array[i] < array[i - 1])

30 localMinIdxs.push_back(i);

31 if (i == 0 || i == array.size() - 1)

32 continue;

33 if (array[i] < array[i + 1] && array[i] < array[i - 1])

1 #include <vector>

2 using namespace std;

3

4 int minRewards(vector<int> scores) {

5 // Write your code here.

6 return -1;

7 }

8

```
1 def test_case_07():
2     assert isPrime(28) == 0
3
4 def test_case_07():
5     assert isPrime(29) == 1
6
7 def test_case_07():
8     assert isPrime(30) == 0
9
10 def test_case_07():
11     assert isPrime(31) == 1
12     assert isPrime(32) == 0
13
14 def test_case_07():
15     assert isPrime(33) == 0
16     assert isPrime(34) == 0
17     assert isPrime(35) == 0
18     assert isPrime(36) == 0
19     assert isPrime(37) == 1
20     assert isPrime(38) == 0
21     assert isPrime(39) == 0
22     assert isPrime(40) == 0
23     assert isPrime(41) == 1
24     assert isPrime(42) == 0
25     assert isPrime(43) == 1
26     assert isPrime(44) == 0
27     assert isPrime(45) == 0
28     assert isPrime(46) == 0
29     assert isPrime(47) == 1
30     assert isPrime(48) == 0
31     assert isPrime(49) == 0
32     assert isPrime(50) == 0
33     assert isPrime(51) == 0
34     assert isPrime(52) == 0
35     assert isPrime(53) == 1
36     assert isPrime(54) == 0
37     assert isPrime(55) == 0
38     assert isPrime(56) == 0
39     assert isPrime(57) == 0
40     assert isPrime(58) == 0
41     assert isPrime(59) == 1
42     assert isPrime(60) == 0
43     assert isPrime(61) == 1
44     assert isPrime(62) == 0
45     assert isPrime(63) == 0
46     assert isPrime(64) == 0
47     assert isPrime(65) == 0
48     assert isPrime(66) == 0
49     assert isPrime(67) == 1
50     assert isPrime(68) == 0
51     assert isPrime(69) == 0
52     assert isPrime(70) == 0
53     assert isPrime(71) == 1
54     assert isPrime(72) == 0
55     assert isPrime(73) == 1
56     assert isPrime(74) == 0
57     assert isPrime(75) == 0
58     assert isPrime(76) == 0
59     assert isPrime(77) == 0
60     assert isPrime(78) == 0
61     assert isPrime(79) == 1
62     assert isPrime(80) == 0
63     assert isPrime(81) == 0
64     assert isPrime(82) == 0
65     assert isPrime(83) == 1
66     assert isPrime(84) == 0
67     assert isPrime(85) == 0
68     assert isPrime(86) == 0
69     assert isPrime(87) == 0
70     assert isPrime(88) == 0
71     assert isPrime(89) == 1
72     assert isPrime(90) == 0
73     assert isPrime(91) == 0
74     assert isPrime(92) == 0
75     assert isPrime(93) == 0
76     assert isPrime(94) == 0
77     assert isPrime(95) == 0
78     assert isPrime(96) == 0
79     assert isPrime(97) == 1
80     assert isPrime(98) == 0
81     assert isPrime(99) == 0
82     assert isPrime(100) == 0
```

Run or submit code when you're ready.