

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 using System;
4 using System.Collections.Generic;
5
6 public class Program {
7     // O(n^2) time | O(n) space
8     public static List<int[]> DiskStacking(List<int[]> disks) {
9         disks.Sort((disk1, disk2) => disk1[2].CompareTo(disk2[2]));
10         int[] heights = new int[disks.Count];
11         for (int i = 0; i < disks.Count; i++) {
12             heights[i] = disks[i][2];
13         }
14         int[] sequences = new int[disks.Count];
15         for (int i = 0; i < disks.Count; i++) {
16             sequences[i] = Int32.MinValue;
17         }
18         int maxHeightIdx = 0;
19         for (int i = 1; i < disks.Count; i++) {
20             int[] currentDisk = disks[i];
21             for (int j = 0; j < i; j++) {
22                 int[] otherDisk = disks[j];
23                 if (areValidDimensions(otherDisk, currentDisk)) {
24                     if (heights[i] <= currentDisk[2] + heights[j]) {
25                         heights[i] = currentDisk[2] + heights[j];
26                         sequences[i] = j;
27                     }
28                 }
29             }
30             if (heights[i] >= heights[maxHeightIdx]) {
31                 maxHeightIdx = i;
32             }
33         }
```

Our Tests

```
1 using System.Collections.Generic;
2
3 public class Program {
4     // O(n^2) time | O(n) space
5     public static List<int[]> DiskStacking(List<int[]> disks) {
6         disks.Sort((disk1, disk2) => disk1[2].CompareTo(disk2[2]));
7         int[] heights = new int[disks.Count];
8         for (int i = 0; i < disks.Count; i++) {
9             heights[i] = disks[i][2];
10         }
11         int[] sequences = new int[disks.Count];
12         for (int i = 0; i < disks.Count; i++) {
13             sequences[i] = Int32.MinValue;
14         }
15         int maxHeightIdx = 0;
16         for (int i = 1; i < disks.Count; i++) {
17             int[] currentDisk = disks[i];
18             for (int j = 0; j < i; j++) {
19                 int[] otherDisk = disks[j];
20                 if (areValidDimensions(otherDisk, currentDisk)) {
21                     if (heights[i] <= currentDisk[2] + heights[j]) {
22                         heights[i] = currentDisk[2] + heights[j];
23                         sequences[i] = j;
24                     }
25                 }
26             }
27             if (heights[i] >= heights[maxHeightIdx]) {
28                 maxHeightIdx = i;
29             }
30         }
```

Solution 1 Solution 2 Solution 3

```
1 using System.Collections.Generic;
2
3 public class Program {
4     public static List<int[]> DiskStacking(List<int[]> disks) {
5         // Write your code here.
6         return null;
7     }
8 }
9
```

Custom Output

Submit Code

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

```

10  return expected == result + 100 * countZeroes
11
12  expected = 1000 * 1000 * 1000 * 1000
13  return countZeroes(expected) == expected
14
15  }
16
17  // Test
18  int main() {
19      countZeroes(1000 * 1000 * 1000 * 1000);
20      countZeroes(1000 * 1000 * 1000 * 1000);
21      countZeroes(1000 * 1000 * 1000 * 1000);
22      countZeroes(1000 * 1000 * 1000 * 1000);
23      countZeroes(1000 * 1000 * 1000 * 1000);
24      countZeroes(1000 * 1000 * 1000 * 1000);
25      countZeroes(1000 * 1000 * 1000 * 1000);
26  }

```

Run or submit code when you're ready.