Statistics Problem Statement

Problem Statement for TheBrickTowerMediumDivOne

Problem Statement

John and Brus just built some towers using toy bricks. They now have n towers numbered 0 through n-1. For each i, the height of the i-th tower (0-based index) is given in **heights**[i].

John and Brus want to arrange their towers into a line. That is, the bottoms of the towers will all stand on the same line. John and Brus don't like it when a tower falls down and knocks over another tower while falling. To avoid this, they have to put their towers sufficiently far apart. More precisely, the distance between any two neighboring towers must be at least equal to the maximum of their heights. John and Brus would like to minimize the distance between the first and the last tower in the line.

You are given the int[] **heights**. Return a int[] containing exactly n elements: the order in which the towers should be placed on the line. For each i, the i-th element of the return value should be the number of the tower that will be placed i-th on the line. If there is a tie (multiple solutions give the same minimal distance), return the lexicographically smallest order.

Definition

```
Class:           TheBrickTowerMediumDivOne
Method:          find
Parameters:      int[]
Returns:         int[]
Method signature:int[] find(int[] heights)
(be sure your method is public)
```

Notes

- A int[] A is lexicographically smaller than a int[] B if it contains a smaller element at the first position where these int[]s differ.

Constraints

- **heights** will contain between 1 and 47 elements, inclusive.
- Each element of **heights** will be between 1 and 47 inclusive.

Examples

0)

{4, 7, 5}
Returns: {0, 2, 1 }
There are six possible orderings, but only four of them have optimal distance 12 between the first and the last towers:

- {0, 2, 1}
- {1, 0, 2}
- {1, 2, 0}
- {2, 0, 1}

Among these orderings {0, 2, 1} is the lexicographically smallest one.

1)

{4, 4, 4, 4, 4, 4, 4}
Returns: {0, 1, 2, 3, 4, 5, 6 }
Towers may have equal heights.

2)

{2, 3, 3, 2}
Returns: {0, 3, 1, 2 }
Towers of height 2 have to be neighboring in the optimal ordering.

3)

{13, 32, 38, 25, 43, 47, 6}
Returns: {0, 6, 3, 1, 2, 4, 5 }

This problem was used for:
Single Round Match 554 Round 1 - Division I, Level Two