Consider the following Java code:

```
 1 boolean isSorted(int[] a) {
 2   return isSortedHelper(a, 0, a.length - 1);
 3 }
 4 boolean isSortedHelper(int[] a, int from, int to) {
 5   if (from > to) {
 6     return true;
 7   }
 8   else if(from == to) {
 9     return true;
10   }
11   else {
12     return a[from] <= a[from + 1]
13       && isSortedHelper(a, from + 1, to);
14   }
15 }
```

Prove, via mathematical induction, that the method `isSorted` method above correctly returns `true` if the array `a` is sorted in a non-descending order; and `false` otherwise.

Solution:

We first prove that the recursive helper method `isSortedHelper` (Line 4 – Line 15) is correct (i.e., is the subarray {`a[from]`, `a[from + 1]`, . . . , `a[to]`} sorted).

1. **Base Cases**
   (a) **Concept**: In an empty array, there is no witness (i.e., adjacent numbers that are not sorted) ∴ result is **true**.
   (b) **Link to Code**: **Lines 5 – 7 (or just Line 6)** of the above code does this.
   (c) **Concept**: In an array of size 1, the only one element is automatically sorted.
   (d) **Link to Code**: **Lines 8 – 10 (or just Line 9)** of the above code does this.
2. **Inductive Cases**
   (a) **Inductive Hypothesis (I.H.)**: The recursive call `isSortedHelper(a, from + 1, to)` returns **true** if a[from + 1], a[from + 2], . . . , a[to] are sorted in a non-descending order; **false** otherwise.
   (b) **Concept**: `isSortedHelper(a, from, to)` should return **true** if:

      1) `a[from]` ≤ `a[from + 1]`; and

      2) the subarray {`a[from + 1]`, . . . , `a[to]`} is sorted.
   (c) **Link to I.H.**: By I.H., condition **2)** is satisfied.
   (d) **Link to Code**: **Line 12** in the above code does condition **1)**.

      ∴ **Lines 12 – Line 13** perform a correct combination.
3. Given that the recursive helper method `isSortedHelper` (Line 4 – Line 4) is correct, we now argue that the method `isSorted` (Line 1 – Line 3) is correct.
   (a) **Concept**: `isSorted(a)` is correct by invoking `isSortedHelper(a, 0, a.length - 1)`, examining the entire array.
   (b) **Link to Code**: **Line 2** of the above code does this.