CPSC 221: Checking Program Correctness for Mergesort

Show that the Mergesort algorithm presented in class, and summarized in the high-level pseudo-code below, is correct. You can assume the correctness of the Merge algorithm.

```
Algorithm Mergesort(A[1..n])     /* pass in an array containing n elements */
    if  (n ≤ 1)
        return A;
    B = Mergesort(A[1..(n/2)])
    C = Mergesort(A[(n/2 + 1)..n])
    return Merge(B, C)                /* merge 2 sorted lists */
```

Note about the Merge algorithm: Since B and C are sorted, we maintain pointers to the first unused element in each array, and simply pick the smaller of the two elements as the next element in our sorted list. Thus, it completes in O(b+c) time, where b and c are the number of elements in each sorted list.

Proof of Correctness of Algorithm Mergesort:

Base Case: $n = 1$

Mergesort correctly returns the single element (which is trivially “sorted”)

Inductive Hypothesis:

Assume that Mergesort correctly sorts n=1, 2, ..., k elements

Inductive Step:

Show that Mergesort correctly sorts $n = k + 1$ elements.

By the Inductive Hypothesis, since `ceiling`(n/2) < k:

B is a sorted array of the first n/2 elements

C is a sorted array of the last `ceiling` (n/2) elements

Then, by the correctness of the Merge algorithm, Mergesort returns the elements in the combined B + C array in sorted order. Furthermore, the resulting array contains all the elements of A (i.e., all $n = k + 1$ elements).

The algorithm terminates because we recursively reduce the size of each array by a factor of 2. Eventually each such array contains a single element, in which case the `if` statement causes Algorithm Mergesort to return. Lastly, we assume that Algorithm Merge terminates; therefore, Mergesort also terminates.