Brookson HW0 - SDS663

2024-09-01

Table of contents

2. For checking an array that is sorted after 0 passes of the algorithm, you could run through the first loop of j and the first loop of k. If there are no swaps made at all, then terminate the algorithm. You could also create a condition where if any k iteration ends in no swaps the algorithm terminates. I don't know how to prove the latter, but the former, if we show for the base case of A being of $n = 1$ (which is already sorted) and no swaps happen, then we can suggest that for $n = 1$, the same will hold true - it is already sorted. If the algorithm passes over (for example) $A = [1, 2]$ and makes no swaps after the first pass, we can reasonably assume for all $n > 2$ that will hold true.
3. I didn't get to this one.
4. The biggest challenge of this was trying to think of how to prove the algorithm without looking up how the induction would work since I don't remember how to do induction on this type of problem, so if I've made a mistake it's probably that I don't sufficiently show correctness of the algorithm in Q1. Also I don't think I'm fully correct in $O(n^2)$, there's definitely another reason it's $O(n^2)$ other than just the number of comparisons but I don't know what it is.

** Resources Used **

- (https://www.programiz.com/c-programming)
- (https://stackoverflow.com/questions/840501/how-do-function-pointers-in-c-work) but didn't end up using anything from this
- (https://w3.cs.jmu.edu/buchhofp/class/cs361_s18/documentation_examples.html)