CS3000: Algorithms & Data — Summer 2023 — Laney Strange

Homework 2 - Short
Due Thursday May 18 @ 9pm Gradescope

Name:
Collaborators:

- Put your name on the first page. If you are using the LaTeX template we provided, then you can make sure it appears by filling in the `yourname` command.
- This assignment is due Thursday May 18 @ 9pm Gradescope. You may submit up to 48 hours late for no penalty, but expect a delay in grading.
- Show ALL your work, even if the problem doesn't specify it.
- You'll have an opportunity to resubmit two homeworks (one long, one short) at the end of the semester.
- Solutions must be typeset, preferably in LaTeX. If you need to draw any diagrams, you may draw them by hand as long as they are embedded in the PDF. I recommend using the source file for this assignment to get started.
- I encourage you to work with your classmates on the homework problems. *If you do collaborate, you must write all solutions by yourself, in your own words.* Do not submit anything you cannot explain. Please list all your collaborators in your solution for each problem by filling in the `yourcollaborators` command.
- If you get stuck on a homework problem, come by office hours or post on Piazza! We recommend you spend about 30 minutes trying to figure out a problem, and then ask for help. We'll be happy to clarify material from class and algorithm concepts, but we will not give out solutions or confirm your answers are correct.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class, is strictly forbidden.

Problem 1. *Mergesort (2 + 2 + 4 = 8 points)*

(a) Is Mergesort in-place, assuming it's implemented the way we saw in class? If not, how much auxiliary space does it require and where is it used?

Solution:

(b) We know that the bound on Mergesort's worst-case run-time is $\Theta(n \lg n)$. What is the bound on its best-case run time?

Solution:

(c) We've used the iteration method to show that the solution to a recurrence like $T(n) = 2T(n/2) + n, T(2) = 2$ is $\Theta(n \lg n)$. Use mathematical induction to show that we have an exact solution: $T(n) = 2T(n/2) + n, T(2) = 2$ gives us $T(n) = n \lg n$, whenever n is a power of 2.

Solution:

Problem 2. *Quicksort (4 + 4 = 8 points)*

(a) Give a recurrence, and its solution/bound, that represents the best-case run-time of Quicksort. You can use any method to solve the recurrence, but cite which one you use and show all your work.

Solution:

(b) Give a recurrence, and its solution/bound, that represents the run-time of Quicksort when all elements in the array have the same value. You can use any method to solve the recurrence, but cite which one you use and show all your work.

Solution:

Problem 3. *More Quicksort! (4 points)*

You can find the pseudocode for our Quicksort implementation at https://course.ccs.neu.edu/cs3000/resources/Quicksort_Pseudocode.pdf. For this problem, we'll focus on showing correctness of the Partition step.

The loop invariant for the loop at lines 3-6 of the Partition step holds that, at the beginning of each iteration of the loop, for any array index k, the following conditions are all true:

- If $p \leq k \leq i$, then $A[k] \leq x$
- If $i + 1 \leq k \leq j - 1$, then $A[k] > x$
- If $k = r$, then $A[k] = x$

For this problem, argue that the first condition of the loop invariant is true.

Solution: