CS256-01: Algorithm Design and Analysis

Assignment 0 (due 02/07/2024)

Instructor: Sam McCauley

Note. This homework will not be graded on correctness but on completion—to get full points, you must attempt all the questions (except the optional feedback question).

The goal of this assignment is for you to check your familiarity with the background material from Data Structures (CS136) and Discrete Math (MATH200), and for you to get comfortable with LaTeX. It is your responsibility to fill in the gaps in your knowledge.

Submission guidelines. When submitting your solution PDF on Gradescope, you must match questions to pages in the PDF. This takes less than a minute and is crucial for efficient and anonymous grading. Sometimes, you may need to mark pages approximately, e.g., for multi-part questions such as Problem 4.

There is one question per page of this assignment. Make sure you scroll to see all pages.

Problem 1. Please complete the intro form; a link has been included below for convenience.

Intro Form: `https://forms.gle/Le3cj5WA7hKF2mwh9`

Problem 2. Let A, B be sets. Prove by contradiction that $A \cap B = \emptyset \implies A \subseteq \overline{B}$.

Solution.

$\square$

Problem 3. The following sorting algorithm appears to be riddled with problems. The inner loop scans the entire array, and it swaps $A[i]$ with $A[j]$ when $A[i] \leq A[j]$—even if $i < j$.

```
TypoSort(A):
    for i = 0 to |A|-1:
       for j = 0 to |A|-1:
           if A[i]<=A[j]: // does not check if i < j
               swap(A[i],A[j])
```

Surprisingly, `TypoSort` correctly sorts an array (try it!). Prove using induction that `TypoSort` is correct.

Hint. Use the following invariant: after k iterations of the outer loop, $A[0]$ through $A[k-1]$ are in sorted order.

You may assume that if $A[0]$ through $A[k-1]$ are already in sorted order, the following algorithm (run on A and k) ensures that $A[0]$ through $A[k]$ are in sorted order:

```
Subroutine(A,k):
   for j = 0 to k:
       if A[k]<=A[j]:
           swap(A[k],A[j])
```

Solution. □

Note. Problem 4 relies on material we will see on Monday, 02/05/23.

Problem 4. For each of the following, answer with the tightest upper bound from this list: $O(\log n)$, $O(n)$, $O(n \log n)$, $O(n^2)$, $O(2^n)$. Briefly justify your answer.

(a) The number of leaves in a complete[1] binary tree of height n:

Solution.

□

(b) The depth of a complete binary tree with n nodes:

Solution.

□

(c) The worst-case run time to sort n items using merge sort:

Solution.

□

(d) The number of distinct subsets of a set of n items:

Solution.

□

(e) The number of bits needed to represent the positive integer n:

Solution.

□

(f) The time to find the second largest number in a set of n (not necessarily sorted) numbers:

Solution.

□

[1]Complete: Every leaf has same depth and every non-leaf node has two children.