CS 3000: Algorithms & Data — Fall 2020

Problem Set 1
Due Tuesday September 22 at 11:59pm via Gradescope

Name:
Collaborators:

- Make sure to 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 Tuesday September 22 at 11:59pm via Gradescope. No late assignments will be accepted. Make sure to submit something before the deadline.
- Solutions must be typeset in LaTeX. If you need to draw any diagrams, you may draw them by hand as long as they are embedded in the PDF. We recommend using the source file for this assignment to get started.
- We encourage you to work with your classmates on the homework problems, but also urge you to attempt all of the problems by yourself first. 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.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class is strictly forbidden.

Problem 1. *(3 + 5 = 8 points) What Does This Code Do?*

You encounter the following mysterious piece of code.

```
Algorithm 1: Mystery Function

Function F(a, n)):
    If n = 0 :
        Return (1, a)
    Else
        b = 1
        For i from 1 to 2n
            b = b · a
        (u, v) ← F(a, n − 1)
        Return (u · b/a, v · b · a)
```

(a) What are the results of $F(a,3)$, $F(a,4)$, and $F(a,5)$. You do not need to justify your answers.

(b) What does the code do in general, when given input integer $n \geq 0$? Prove your assertion by induction on n. F(a,n) = (a^n^2, a^(n+1)^2)

Problem 2. *(5 + 5 = 10 points) Recursion and induction in binary codes*

Digital transmission protocols transmit signals using binary codes. In order to minimize the effect of errors, protocols often use codewords for signals with the property that "similar" signals use "similar" codewords.

One such code is a list of 2^n n-bit strings in which each string (except the first) differs from the previous one in exactly one bit. Let us call such a list a bit-flip list since we go from one string to the next by just flipping one bit.

Consider the following recursive algorithm for listing the n-bit strings of one such bit-flip list.

- If $n = 1$, the list is 0,1.
- If $n > 1$, first take a bit-flip list of $(n-1)$-bit strings, and place a 0 in front of each string. Then, take a second copy of the same bit-flip list of $(n-1)$-bit strings, place a 1 in front of each string, reverse the order of the strings and place it after the first list.

For example, for $n = 2$, the list is 00,01,11,10, and for $n = 3$, we get 000,001,011,010,110,111,101,100.

Prove the following two statements by induction on n.

(a) Every n-bit string appears exactly once in the list generated by the algorithm.

(b) Each string (except the first) differs from the previous one in exactly one bit.

Problem 3. *(7 points) Reconstructing a total order*

A group of n runners finished a close race. Unfortunately, the officials at the finish line were unable to note down the order in which the racers finished. Each runner, however, noted the jersey number of the runner finishing immediately ahead of her or him. (There were no ties.)

The race officials ask each runner to give an ordered pair, containing two pieces of information: (a) first, his or her own jersey number and (b) second, the jersey number of the runner who finished immediately ahead of him or her. The winner of the race, who did not see anybody finish ahead of her, enters $\perp$ for (b).

You have been asked to design an algorithm that takes as input the n pairs and returns the order in which the runners finished the race. Assume each runner is honest.

Give a deterministic $\Theta(n \log n)$ time algorithm, and justify the running time of the algorithm. (Hint: Use sorting.)

Problem 4. *(5 points) Use induction to prove that logarithmic functions are "smaller" than polynomials*

In class we assert without proof that "logarithmic functions are smaller than polynomials." Specifically, that $(\log_2 n)^a = O(n^b)$ for every $a > 0$ and $b > 0$. In this problem you will use induction to prove a special case of this fact, that $\log_2 n = O(\sqrt{n})$.

First prove by induction that, for every ~~$k \geq 2$~~ $k \geq 4$, $\log_2(2^k) \leq 2^{k/2}$. Then, prove that $\log_2 n \leq \sqrt{2n}$ for all $n \geq 1$ by using the fact that $\log_2 n$ is at most $\log_2(2^k)$ for the smallest integer k such that $n \leq 2^k$.

Problem 5. *($3 \times 4 = 12$ points) Growth of functions*

For each of these parts, indicate whether $f = O(g)$, $f = \Omega(g)$, or both (i.e., $f = \Theta(g)$). In each case, give a justification for your answer. Your justification can be in the form of a proof from first principles or a proof using limits, and can use any of the facts presented in the lecture or the text. (Hint: It may help to plot the functions and obtain an estimate of their relative growth rates. In some cases, it may also help to express each function as a power of 2 and then compare.)

(a) $f(n) = n2^n$; $g(n) = 3^n$.

(b) $f(n) = n \log n$; $g(n) = n^{4/3}$.

(c) $f(n) = 10n + (\log n)^2$; $g(n) = 100n - 8\sqrt{n}$.

Problem 6. *($4 + 4 = 8$ points) Properties of asymptotic notation*

Let $f(n)$ and $g(n)$ be asymptotically positive and monotonically increasing functions.

(a) Define $h(n) = \max\{f(n), g(n)\}$, for all $n \geq 0$. Prove that $f(n) + g(n) = \Theta(h(n))$.

(b) Let a be an arbitrary positive real number. Prove that if $f(n) = O(g(n))$, then $f(n)^a$ is $O(g(n)^a)$.