Homework 1 Solutions: CSCI 6212 Algorithms, Fall 2019

Grading Rubric

Homework was due Beginning of class, September 20, 2019. Class the day it was turned in started at 10:30am. Submissions before 10:40 am were given full credit. -2 points (out of 20) if they were late. Homeworks submitted after midnight on September 20, 2019 were given 0 points. Please contact the professor in this case.

1. Sort the following functions from asymptotically smallest to asymptotically largest, indicating ties if there are any. Do not submit proofs, just a sorted list of 16 functions.

   Write $f(n) \prec g(n)$ to indicate that $f(n) = o(g(n))$, and write $f(n) \equiv g(n)$ to mean $f(n) = \Theta(g(n))$. We use the notation $lg(n) = log_2(n)$.

n	$\lg n$	$\sqrt{n}$	7^n
$\sqrt{\lg n}$	$\lg \sqrt{n}$	$7^{\sqrt{n}}$	$\sqrt{7^n}$
$7^{\lg n}$	$\lg(7^n)$	$7^{\lg \sqrt{n}}$	$7^{\sqrt{\lg n}}$
$\sqrt{7^{\lg n}}$	$\lg(7^{\sqrt{n}})$	$\lg \sqrt{7^n}$	$\sqrt{\lg(7^n)}$

   A *wrong* answer, in the correct format, might look like:

$$n \prec lg\ n \equiv 7^{\sqrt{n}} \equiv 7^{lg\ \sqrt{n}} \prec \sqrt{lg\ n} \ldots$$

   This problem was worth 6 points. The score is max(6 - number of things in the wrong order, 3 points if "right idea but several mistakes , otherwise (1 point for correct minimum + 1 point for correct maximum)

   Correct answer (Needs work!)

$$\sqrt{lgn} \prec$$
$$lgn \equiv lg\sqrt{n} \prec$$
$$7^{\sqrt{lgn}} \prec$$
$$\sqrt{n} \equiv lg(7^{\sqrt{n}}) \equiv \sqrt{lg(7^n)} \prec$$
$$n \equiv lg(7^n) \equiv lg(\sqrt{7^n}) \prec$$
$$\sqrt{7^{lgn}} \equiv 7^{lg\sqrt{n}} \prec$$
$$7^{lgn} \prec$$
$$7^{\sqrt{n}} \prec$$
$$\sqrt{7^n} \prec$$
$$7^n$$

2. The following are the time complexity for 4 different functions. For each, solve the recurrence and give them complexity in Θ notation

   (a) $T(n) = 3T(\frac{n}{3}) + 1$
   (b) $T(n) = 3T(\frac{n}{3}) + n$
   (c) $T(n) = 6T(\frac{n}{3}) + n$

   Two points each

   (a) $\theta(n)$
   (b) $\theta(nlog(n))$
   (c) $\theta(n^{\log_3 6})$

3. Like many of the algorithms we will describe this semester, our presentation of the GaleShapley (GS) algorithm was very high-level. As competent programmers, I will usually assume that you can add the necessary (hopefully easy) implementation details. For the sake of concreteness, lets consider how we would do that for this algorithm.

   (a) Consider the pseudo-code below for the GS algorithm. Describe what data structures (lists, arrays, stack, queues, hash tables, etc.) you would use for implementing the code below.
   (b) Using the data structures from part (a), explain how to implement that GS algorithm so that it runs in $O(n^2)$ time, where n is the number of men and the number of women in the system.

```
Initially all men and all women are unengaged
while (there is an unengaged man who hasnt yet proposed to every woman) {
    Let m be any such man
    Let w be the highest woman on his list to whom he has not yet proposed
    if (w is unengaged) then (m, w) are now engaged
    else {
         Let m be the man w is engaged to currently
         if (w prefers m to m) {
         Break the engagement (m, w)
         Create the new engagement (m, w)
         (m is now unengaged)
         }
     }
}
```

(a) 4 points for clearly describing what data structures are used. Needs to include:
1. Exist data structure
2. Their prefer orders
3. Their status of proposed
4. Their status of engaged

how the algorithm remembers who is engaged to who (is it an array? a linked list?) How does the algorithm find an unengaged man? How does the algorithm find the most preferred woman that they have not yet proposed to.

(b) Need to give an argument about why these data structures can be used to give a total $O(n^2)$ runtime.
1. Correctness (2 points)
2. Making argument about runtime of each important line. (1 point)
3. Good choice of data structure(1 point)

```
Example solution:
M: Array nxn of men's choices, where M(i,j) is the "j-th" favorite
   woman for man i.
W: Array nxn of women's choices, where W(i,j) is the "j-th" favorite
    man for woman i.
C: Array nx1 of how many women each man has proposed to.

EM(i) is the women to whom man i is currently engaged.
EW(i) is the man to whom woman i is currently engaged.

With these arrays, every step of the above algorithm can be
done in constant time.

With these data structures and a few extra operations to keep
them updated, all operations in the loop can be done in constant time.

Loop executes a total of n^2 time, so these data structures suffice.
```