**CSCI 2834-003/004: Discrete Structures Workgroup – Fall 2019** **Worksheet 07**

Get in small groups (about 4 students maximum) and work out these problems on the whiteboard. Ask one of the teaching assistants for help if your group gets stuck. You do **not** need to turn anything in. Since this worksheet contains a lot of problems, a good strategy would be to first skim the worksheet and then discuss and solve the problems which you think are difficult.

## Cardinality

1. Show that if $A$ and $B$ are finite sets, then $|A \cap B| \leq |A \cup B|$. Determine when this relationship is an equality.
2. Let $A$ and $B$ be subsets of the finite universal set $U$. Show that $|\overline{A} \cap \overline{B}| = |U| - |A| - |B| + |A \cap B|$
3. Suppose that $f$ is a function from A to B where A and B are finite sets. Explain why $|f(S)| = |S|$ for all subsets S of A if and only if $f$ is one-to-one.

## Recursion

4. Find a closed-form solution for the following recurrence relations:
   (a) $a_n = a_{n-1} + 1 \mid a_0 = 2$
   (b) $a_n = -3a_{n-1} \mid a_0 = 7$
   (c) $a_n = 4a_{n-2} \mid a_0 = 2$
   (d) $a_n = (a_{n-1})^n \mid a_0 = -3$
   (e) $a_n = \frac{-a_{n-1}}{n} \mid a_1 = -4$
   (f) $a_n = n - a_{n-1} \mid a_0 = 5$

5. Conjecture a simple formula for $a_n$ if the first 5 terms of the sequence $a_n$ are 1, 7, 25, 79, 241.

6. Find a closed form for the sequence $\{a_n\}_{n=0}^{\infty}$ defined by:

$$\frac{1}{1 \cdot 2} + \frac{1}{2 \cdot 3} + \ldots + \frac{1}{n(n+1)}$$

## Algorithms

7. Suppose you have an algorithm that you have been making timing measurements using a variety of different input sizes. For each case below, the table provides the size of the input to the algorithm, and the amount of time (in seconds) required for the algorithm to execute with the provided input. In each case, determine the complexity of the algorithm.

(a)

| Input Size | Time |
|---|---|
| 2 | 5 |
| 4 | 20 |
| 8 | 80 |

(b)

| Input Size | Time |
|---|---|
| 3 | 1 |
| 6 | 2 |
| 12 | 4 |
| 24 | 8 |

(c)

| Input Size | Time |
|---|---|
| 10 | 2 |
| 20 | 16 |
| 40 | 128 |

8. For the following problems, figure out an algorithm, then write out Python pseudocode on paper/whiteboard. If there is time left, or later in the comfort of your own home, code up the solutions!

   (a) Given two input strings, write a program that finds the longest common subsequence.

   (b) Write a function that merges two sorted lists into a new sorted list. $[1, 4, 6], [2, 3, 5] \rightarrow [1, 2, 3, 4, 5, 6]$. Note that this can be done more efficiently than simply concatenating the two lists and then sorting the larger list from scratch.