CSCE 222: Discrete Structures for Computing
Section 502 & 503
Fall 2020

YOUR NAME HERE

Homework 3

Due: 27 September 2020 (Sunday) before 11:59 p.m. on Gradescope.
You must show your work in order to recieve credit.

Aggie Honor Statement: On my honor as an Aggie, I have neither given nor received any unauthorized aid on any portion of the academic work included in this assignment.

Checklist: Did you...
1. abide by the Aggie Honor Code?
2. solve all problems?
3. start a new file for each problem?
4. show your work clearly?
5. submit the PDFs (and programming language source code, if applicable) to Gradescope?

Problem 1.

Part a. Let A and B be two sets. Suppose that:
1. $A - B = \{1, 5, 7, 8\}$
2. $B - A = \{3, 6, 9\}$
3. $A \cap B = \{1, 5, 7, 8\}$

What are the sets A and B?

Part b. Let A, B and C be three sets. Suppose $A \cup C = B \cup C$. Can you conclude $A = B$? If your answer is "yes", prove it. If your answer is "no", give a counterexample.

Problem 2. In section 2.4.3, the topic is a special type of sequence called a *recurrence relation*. We'll discuss these in more detail in Chapter 8.

Find the first six terms of the sequence defined by each of these recurrence relations and initial conditions:
1. $a_n = 2a_{n-1}, a_0 = 1$
2. $a_n = -2a_{n-1}, a_0 = -1$
3. $a_n = a_{n-1} + a_{n-2}, a_0 = 0, a_1 = 1$
4. $a_n = 2a_{n-1}^2, a_0 = 1$

Problem 3. Consider the matrices

$$A = \begin{bmatrix} 3 & -1 \\ -2 & 4 \end{bmatrix}, B = \begin{bmatrix} 1 & 2 \\ 3 & -1 \end{bmatrix}, \text{ and } C = \begin{bmatrix} 1 & 0 \\ 2 & 3 \end{bmatrix}, \text{ find}$$

1. AB
2. BA
3. $(AB)C$ (Show the results of all intermediate matrix multiplications.)
4. $A(BC)$ (Show the results of all intermediate matrix multiplications.)

5. What observations do you make about the commutivity and associativity of matrix operations? (You are not proving these properties, just expressing what you see for this one set of matrices.)

Problem 4. Section 2.2.4 tells how to represent sets in a computer using bit strings. In this way, a single 32-bit integer (the type of **int** used in most current implementations of C and all Java implementations) could be used to represent a finite universe containing up to 32 possible elements. The union of two sets is done with a bitwise OR (|), the intersection of two sets with a bitwise AND (&) and the complement of two sets with the NOT operator ($\sim$).

With this in mind, it's easy to see that a single integer can represent a set of letters from the English alphabet {A, B, ..., Z} by mappinng A...Z to 0...25 and using to represent the corresponding bit of the integer. Write a program (in the programming language of your choice) to solve the following problem:

> For each sentence of all uppercase letters that is received in a string, check to see if all 26 letters of the alphabet are present at least once. If not print a list of the missing letters. (A sentence that does contain all 26 alphabetic characters is called a *pangram*.)

Here's a potential algorithm:

> Start with a null set and, using the set union operation, expand it by adding to it each incoming character. (If the character is already a member, the union operation actually does nothing, which is just what we want.) When this step is done, see if the set is complete. If it is not, print the missing letters – that is, the difference between it and the set {A, B, ... Z}.