For each problem, please indicate how much time you spent on it.

Problem 1
Let $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \cup \{+, \times, -, \div\}$ be an alphabet.
A string $w \in \Sigma^*$ is called a simple numeric expression if it is in one of the following two forms.

1. $w$ consists of one or more digits with no leading zeros.
2. $w = w_1 \otimes w_2$ where $w_1$ and $w_2$ are also simple numeric expressions and
   $\otimes \in \{+, \times, -, \div\}$.

Therefore 1074 is a simple numeric expression of the first kind; $50 + 7$ is a simple numeric expression of the second kind; and $2 - 700 \times 8 \div 10$ is also a simple numeric expression because of its recursive definition.

Let $S \subseteq \Sigma^*$ be the set of all such simple numeric expressions. We also note that $0 \in S$ because 0 is a digit without any leading zeros.

a) Give a regular expression that recognizes $S$.

b) Give a DFA or an NFA that recognizes $S$.

Problem 2
Let $\Sigma$ be the alphabet from Problem 1, and let $\widehat{\Sigma} = \Sigma \cup \{(, )\}$ (thus $\widehat{\Sigma}$ contains all the elements of $\Sigma$ along with the left- and right-parentheses). A string $w \in \widehat{\Sigma}^*$ is called a numeric expression if it is in one of the following two forms.

1. $w$ is a simple numeric expression (i.e., of the form in problem 1), or
2. $w = (x)$ where $x \in \widehat{\Sigma}^*$ is also a numeric expression.

Note that all simple numeric expressions are trivially numeric expressions (because of the first form). However, we can now have expressions with parentheses such as $(2 - 700) \times 8 \div 10$ and $((1 + 2) \times (3 + 4))$.

Let $N \subseteq \widehat{\Sigma}^*$ be the set of all such numeric expressions. Prove that $N$ is not regular using the pumping lemma.
Problem 3
Let \( \Sigma \) be an alphabet and let \( w \in \Sigma^* \) be a string. We write \( w^R \) to denote the reverse of \( w \), i.e., if
\[
w = a_1a_2a_3 \cdots a_n
\]
where \( a_i \in \Sigma \) for each \( 1 \leq i \leq n \), then
\[
w^R = a_na_{n-1} \cdots a_1.
\]
If \( A \subseteq \Sigma^* \) is a language, we also define the reverse of \( A \) to be
\[
A^R = \{ w^R \mid w \in A \}.
\]
Prove that the class of regular languages is closed under the reverse operator.

Problem 4
Let \( E \) be the language of all evenly lengthed bit strings. Therefore, \( 0111 \in E \), but \( 000 \notin E \). (Note that \( \epsilon \in E \) because we count 0 as an even number.)
a) Prove or disprove that \( E \circ E^R \) is regular.
b) Prove or disprove that \( \hat{E} = \{ ww^R \mid w \in E \} \) is regular.
c) Explain in your own words the difference between the languages \( E \circ E^R \) and \( \hat{E} \) and why one is regular and the other is not.
(Note that if you give a disproof, you must use the pumping lemma.)

Bonus Problem: Prefixes
We say a string \( x \in \Sigma^* \) is a prefix of a string \( y \in \Sigma^* \), and we write \( x \sqsubseteq y \), if \( y = xz \) for some string \( z \in \Sigma^* \).

We define the set operators \( P_1 \) and \( P_2 \) by
\[
P_1(A) = \{ x \in \Sigma^* \mid y \sqsubseteq x \text{ for some } y \in A \}
P_2(A) = \{ x \in \Sigma^* \mid x \sqsubseteq y \text{ for some } y \in A \}
\]
for all languages \( A \subseteq \Sigma^* \).
a) Prove or disprove that the set of regular languages is closed under \( P_1 \).
b) Prove or disprove that the set of regular languages is closed under \( P_2 \).
Bonus Problem: Unambiguous Syntax

Let \( G = (V, \Sigma, R, S) \) be a context free grammar defined by

\[
V = \{ \langle \text{STMT} \rangle, \langle \text{IF-THEN} \rangle, \langle \text{IF-THEN-ELSE} \rangle, \langle \text{ASSIGN} \rangle \}, \\
\Sigma = \{ \text{if, condition, then, else, a:=1} \}, \\
S = \langle \text{STMT} \rangle,
\]

and where \( R \) consists of the following rules

\[
\langle \text{STMT} \rangle \rightarrow \langle \text{ASSIGN} \rangle | \langle \text{IF-THEN} \rangle | \langle \text{IF-THEN-ELSE} \rangle \\
\langle \text{IF-THEN} \rangle \rightarrow \text{if condition then} \langle \text{STMT} \rangle \\
\langle \text{IF-THEN-ELSE} \rangle \rightarrow \text{if condition then} \langle \text{STMT} \rangle \text{ else} \langle \text{STMT} \rangle \\
\langle \text{ASSIGN} \rangle \rightarrow \text{a:=1}
\]

a) Show that \( G \) is ambiguous by giving two derivations of the same string. Explain why this could be a serious error in a programming language parser.

b) Give an equivalent grammar to \( G \) that is unambiguous. (You do not need to formally prove that it is unambiguous, but you must explain why you believe it fixes the issue.)