Problem 1. [10 points] Construct a regular expression $R$ over the alphabet $\{0, 1\}$ such that $L(R)$ is the set of words that do not contain 100 as three consecutive characters.
Problem 2. [15 points] Let

$$
\Sigma = \left\{ \begin{bmatrix} a \\ b \\ c \end{bmatrix} : a, b, c \in \{0, 1\} \right\}.
$$

A string \( w \in \Sigma^* \), which can be written as

$$
\begin{bmatrix} a_{n-1} \\ b_{n-1} \\ c_{n-1} \end{bmatrix} \cdots \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix} = \prod_{i=n-1}^{0} \begin{bmatrix} a_i \\ b_i \\ c_i \end{bmatrix},
$$

encodes three natural numbers

$$
a_w = \sum_{i=0}^{n-1} 2^i a_i, \quad b_w = \sum_{i=0}^{n-1} 2^i b_i \quad \text{and} \quad c_w = \sum_{i=0}^{n-1} 2^i c_i.
$$

Prove that the language \( \text{PLUS} = \{ w \in \Sigma^* : a_w + b_w = c_w \} \) is regular. If you utilize DFA, NFA or regular expressions, you may simply state their language instead of proving they describe that language (but you need to formally define them).
Problem 3. [20 points] The substrings of a string are the strings that can be obtained by erasing characters from the original string. This can be defined inductively as follows, where \( \Sigma \) is the strings’ alphabet: \( \varepsilon \) is a substring of any string; and, for \( a \in \Sigma \) and \( w \in \Sigma^* \), the string \( aw \) is a substring of a string \( w' \in \Sigma^* \) if we can write \( w' = xay \) with \( x, y \in \Sigma^* \) and \( w \) a substring of \( y \).

Prove that, whenever \( A \subseteq \Sigma^* \) is a regular language, so are the languages

\[
\text{SUB}(A) = \{ w \in \Sigma^* : w \text{ is a substring of a string in } A \}
\]

and

\[
\text{SUPER}(A) = \{ w \in \Sigma^* : \text{there is a substring of } w \text{ in } A \}.
\]

If you utilize DFA, NFA or regular expressions, you may simply state their language instead of proving they describe that language (but you need to formally define them).
Problem 4. [10 points] Let $A$ be a finite language over an alphabet $\Sigma$. We have seen that $A$ is regular. Prove that any DFA $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = A$ satisfies $|Q| > |w|$ for each string $w \in A$. 

Problem 5. [20 points] Recall our scheme to encode DFA as strings over an alphabet $\Sigma$ with $|\Sigma| \geq 2$. Prove that the language

$$\text{EQ}_{\text{DFA}} = \{ (M_0, M_1) : M_0 \text{ and } M_1 \text{ are DFA such that } L(M_0) = L(M_1) \}$$

is decidable. If you utilize DFA, NFA or regular expressions, you may simply state their language instead of proving they describe that language (but you need to formally define them). If you utilize Turing machines, a high-level description of them is sufficient.

Hint: start by showing that the language

$$\text{NULL}_{\text{DFA}} = \{ (M) : M \text{ is a DFA such that } L(M) = \{ \} \}$$

is decidable. Then reduce the decidability of $\text{EQ}_{\text{DFA}}$ to that of $\text{NULL}_{\text{DFA}}$. 
Problem 6. [15 points] Assume we agree on a scheme analogous to that of Problem 5 to encode context-free grammars as strings over an alphabet $\Sigma$ with $|\Sigma| \geq 2$. Consider then the language

$$\text{EQ}_{\text{CFG}} = \{(G_0, G_1): G_0 \text{ and } G_1 \text{ are context-free grammars such that } L(G_0) = L(G_1)\},$$

which is known to be undecidable (after the exam is over, you can see a proof in Chapter 5 of Sipser’s book).

Assuming this fact, prove or disprove that $\text{EQ}_{\text{CFG}}$ is enumerable. If you utilize Turing machines, a high-level description of them is sufficient.
Problem 7. [10 points] Develop a $\lambda$-term $\text{Reverse}$ such that, whenever a $\lambda$-term $L$ normalizes to a list, the $\lambda$-term $(\text{Reverse } L)$ normalizes to the reverse of that list.