Chapter 1, problems 12, 22.
For problem 12, be sure to explain what is the error in reasoning in the proof. It is not good enough to say that something must be wrong.

Chapter 2, problem 10 parts ii, v; problem 11 parts i, iii.

5. Let \( w = a_1a_2 \ldots a_n \) be a string over the alphabet \( \Sigma \). \( w^R \), the reversal of \( w \), denotes the string \( a_n a_{n-1} \ldots a_1 \). e.g. if \( w = abcc \) then \( w^R = ccba \).

Let \( r \) be a regular expression, and let \( L(r) \) be the language generated by \( r \). Define \( L^R(r) = \{ w \mid w^R \in L(r) \} \) to be the reversal of \( L(r) \), i.e. \( L^R(r) \) contains the reverse of strings in \( L \).

Give a regular expression to generate \( L^R(r) \).

Hint. You will need to proceed inductively using strong induction. The base cases will be for regular expressions defined by the first three rules for creating regular expressions, and the inductive cases will be for regular expressions defined by the remaining three rules.

Challenge problem. This is for those who want an additional challenge. It is NOT for extra credit and will have no bearing on your final grade. However, if you are considering asking me for a letter of recommendation for a PhD program it would not hurt to attempt the challenge problems I will be giving on occasion. Please let me know if you submit such a problem so that I know to look for it.

Chapter 1, Problem 32.
If this is not challenging enough, try Problem 33 (the solution I have in mind uses \( n \) helper envelopes and one “super-envelope” which can hold the \( n \) helper envelopes).