1. Give a regular expression for the following languages over the alphabet \{ a, b, c \}:
   (a) strings where every third symbol is a
   (b) strings where the number of a’s is divisible by three
   (c) strings where all the symbols are the same
   (d) strings that contain at least two different symbols
   (e) strings that contain the substring abc
   (f) strings that do not contain the substring abc.

2. (Sipser Exercise 1.22) Define a comment as a string that begins with the two characters "/*", ends with
   the two characters "/*" and does not contain a */ combination otherwise. For simplicity we consider
   comments consisting of only characters 'a', 'b', '+' and '/'. Give a (a) DFA (b) regular expression for
   the language that consists of all comments.

3. Find out how regular expressions can be used in Java (or a programming language of your choice).
   Using regular expressions, write a program that asks for an input string and then checks whether the
   input string is
   (a) a bit string (consists of characters 0 and 1)
   (b) a time of day in format hh:mm:ss
   (c) a student number (begins with 01 followed by 7 digits)

4. Create a DFA for the language \((0 ∪ 01)^*\) in alphabet \{ 0, 1 \}. Try to make the DFA as simple as possible.
   Then construct an NFA for the same language, using the construction for converting a regular expression
   into an NFA (proof of Lemma 1.55 in Sipser’s book). Then convert your NFA into a DFA using the
   procedure from Sipser’s Theorem 1.39. Compare the two DFAs with each other.

5. Suppose we are given NFAs \(N_1\) and \(N_2\), with \(L(N_1) = A\) ja \(L(N_2) = B\). Assume in addition that
   both NFAs have just one accept state. Consider the following attempts to construct an NFA for a new
   language:
   (a) we try to get an NFA for \(A \cup B\) by combining the start states of \(A\) and \(B\)
   (b) we try to get an NFA for \(A \circ B\) by combining the accept state of \(A\) with the start state of \(B\)
   (c) we try to get an NFA for \(A^*\) by adding an \(ε\) transition from the accepting state to the start state and
       making the starting state into an accept state.
   These are all attempts to simplify the constructions given in the text book by leaving out states and \(ε\)
   transitions. Write these constructions out formally, as in the proofs of Theorems 1.45–1.48 of the text
   book. Which of the constructions (if any) are correct, i.e., always give an NFA for the desired language.
   For the incorrect ones, give a counter example where the construction fails.

6. (a) Let \(A\) be the language represented by the regular expression \(a^*(b ∪ cba)^*(b ∪ c)^*\). Give a regular
       expression for the reverse language \(A^R\) (as defined in Homework 4.4).
   (b) More generally, let \(E\) be a regular expression for a language \(L\). Show that there is a regular expres-
       sion for \(L^R\).