1. Let $L = \left\{ w \in \{\mathtt{0}, \mathtt{1}\}^* \mid w \text{ starts and ends with } \mathtt{0} \right\}$.

   (a) Construct an NFA for L with exactly three states.

   (b) Convert the NFA you just constructed into a DFA using the incremental subset construction. Draw the resulting DFA. Your DFA should have four states, all reachable from the start state.

   (c) Convert the DFA you constructed in part (b) into a regular expression using the state elimination algorithm.

   (d) Write a simpler regular expression for L.

2. (a) Convert the regular expression $(\mathtt{0}^*\mathtt{1} + \mathtt{01}^*)^*$ into an NFA using Thompson's algorithm.

   (b) Convert the NFA you just constructed into a DFA using the incremental subset construction. Draw the resulting DFA. Your DFA should have four states, all reachable from the start state. (Some of these states are obviously equivalent, but keep them separate.)

   (c) Convert the DFA you constructed in part (b) into a regular expression using the state elimination algorithm. You should *not* get the same regular expression you started with.

   (d) **Think about later:** Find the smallest DFA that is equivalent to your DFA from part (b) and convert that smaller DFA into a regular expression using the state elimination algorithm. Again, you should *not* get the same regular expression you started with.

   (e) What is this language?

3. An all-NFA M is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ that accepts $x \in \Sigma^*$ if **every** possible state that M could be in after reading input x is a state from F. Note, this is in contrast to an ordinary NFA that accepts a string if some state among these possible states is a an accept state. Prove that all-NFAs recognize the class of regular languages.

4. Let $L = \left\{ w \in \{\mathtt{0}, \mathtt{1}\}^* \mid \text{a } \mathtt{0} \text{ appears in some position } i \text{ of } w, \text{ and a } \mathtt{1} \text{ appears in position } i+2 \right\}$.

   (a) Construct an NFA for L with exactly four states.

   (b) Convert the NFA you just constructed into a DFA using the incremental subset construction. Draw the resulting DFA. Your DFA should have eight states, all reachable from the start state.

   (c) Convert the NFA you constructed in part (a) into a regular expression using the state elimination algorithm.

5. **Think about later:**

   (a) Convert the regular expression $(\varepsilon + (\mathtt{0} + \mathtt{11})^*\mathtt{0})\mathtt{1}(\mathtt{11})^*$ into an NFA using Thompson's algorithm.

   (b) Convert the NFA you just constructed into a DFA using the incremental subset construction. Draw the resulting DFA. Your DFA should have six states, all reachable from the start state. (Some of these states are obviously equivalent, but keep them separate.)

(c) Convert the DFA you constructed in part (b) into a regular expression using the state elimination algorithm. You should *not* get the same regular expression you started with.

(d) What is this language?