1. For these problems, we will denote by Σ the regular expression $0 + 1$. If x is a regular expression, we will denote $x^0 = \varepsilon$ and $x^k = xx^{k-1}$ as usual.

   (a) $(\varepsilon + \Sigma + \Sigma^2 + 1\Sigma^2 + 01\Sigma + 000) + \Sigma^4\Sigma^*$.

   The $2^0 + 2^1 + 2^2 + 2^3 - 1 = 14$ strings of length ≤ 3 are covered by the elements in the first set of parentheses in the top level union. Strings of length ≥ 4 are then covered by $\Sigma^4\Sigma^*$.

   (b) $\Sigma^*(001)^2$.

   We can have any set of prefixing characters, but we must end with two sets of 001.

   (c) $\Sigma^*001\Sigma^*$.

   We can have any prefix and any suffix, as long as 001 is somewhere in the middle.

   (d) $\Sigma^*0\Sigma^*0\Sigma^*1\Sigma^*$.

   We must have two zeroes before a one, but we can have any number of characters in between.

   (e) $(1 + 01)^*0^*$.

   $(1 + 01)^*$ describes the set of strings that end in 1 that do not contain 001. Namely, we are only allowed to have up to one 0 before we see a 1. However, if the string ends in 0, we can have any number of zeroes at the end because we never form the substring 001 there, so we attach the 0^*.

   (f) We split the strings into cases as follows.

   (1) Strings that have no 0's preceeding 1's. This corresponds to 1^*0^*.

   (2) Strings that have precisely one 0 which is succeeded by 1's. This corresponds to $1^*01^*0^*$.

   If there are two 0's succeeded by 1's, we get the subsequence 001. We can then reduce the regular expression $1^*0^* + 1^*01^*0^* = 1^*(\varepsilon + 01^*)0^*$.

2. Let $Q = \{0,1\}^4 \times \{\varepsilon, 0, 1\}$ and $\Sigma = \{0,1\}$. Very informally, $Q \ni q = (a_{00}, a_{01}, a_{10}, a_{11}, \sigma)$ can be described as $a_s = 1$ if s has been found as a substring and $a_s = 0$ if s has not been found as a substring, and σ is the last seen input symbol. We define $\{(1,1,1,1,\varepsilon), (1,1,1,1,0), (1,1,1,1,1)\}$ as our accepting states and $q_0 = (0,0,0,0,\varepsilon)$ as our start state. We will give a transition function $\delta : Q \times \{0,1\} \to Q$ as follows.

q	σ	$\delta(q,\sigma)$
$(a_{00}, a_{01}, a_{10}, a_{11}, \varepsilon)$	σ	$(a_{00}, a_{01}, a_{10}, a_{11}, \sigma)$
$(a_{00}, a_{01}, a_{10}, a_{11}, 0)$	0	$(1, a_{01}, a_{10}, a_{11}, 0)$
$(a_{00}, a_{01}, a_{10}, a_{11}, 0)$	1	$(a_{00}, 1, a_{10}, a_{11}, 1)$
$(a_{00}, a_{01}, a_{10}, a_{11}, 1)$	0	$(a_{00}, a_{01}, 1, a_{11}, 0)$
$(a_{00}, a_{01}, a_{10}, a_{11}, 1)$	1	$(a_{00}, a_{01}, a_{10}, 1, 1)$

This DFA can be thought as:

(1) If we haven't seen a character yet and we see σ, update the last seen character to σ.

(2) Examine the last seen character and the current character and mark the corresponding substring as "seen".

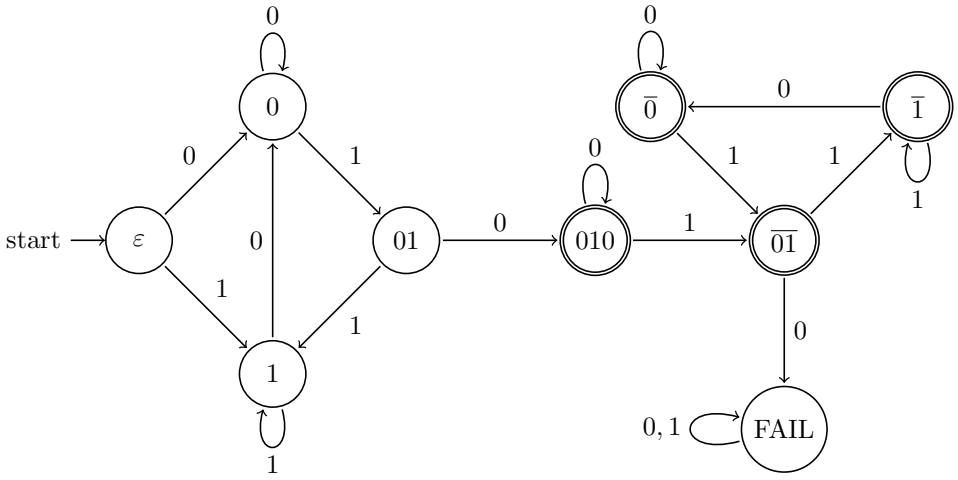
Several states are clearly never reached (namely the ones with ε and not all the trackers are set to 0), but we can just delete those states to get another equivalent DFA.

3. (a) We give a regular expression derived from the DFA we find in (b). In particular, we take a union of how to arrive at each accepting state. The regular expression is as follows.

$$(0 + 1^+0)(0 + 11^+0)^*10^+[\varepsilon + 1(1^+0^+1)^* + 11(1 + 0^+11)^* + 11^+0(0 + 11^+0)^*].$$

The initial $0 + 1^+0$ describes how to initially get to the 0 state and $(0 + 11^+0)^*$ accounts for the possible cycles at 0. To accept, we must then take a 1 and then some positive amount of 0's, which gets us to the 010 state. Since all accepting states must have this as a prefix, we then split up the possible ending states into stopping at 010, advancing to $\overline{01}$ and traversing the cycle some number of times, advancing to $\overline{1}$ and cycling for some number of times, and advancing to $\overline{0}$ and cycling for some number of times. For example, to get to $\overline{1}$ from 010, we must read 11. Then there are two cycles. One is given by 1 and the other by 11^+0, and both can be taken any number of times, hence the Kleene star. Similar logic applies to the other accepting states.

(b)



We can reason about this DFA as follows. Everything to the left of the initial accepting 010 state will keep help us locate the first occurrence of 010. Namely, the only way to reach the state 010 is by taking an edge labelled 0, then 1, then 0. If we are "interrupted" along our path by starting with a 1 or receiving a 1 after a 01, we reset to the 1 state (to minimize this side of the DFA, we can also set $\varepsilon = 1$). The right side of the initial accepting 010 state is similar. We keep track of whenever we reach another 01 by resetting to $\overline{1}$ if we get extra 1's, and if we obtain another 0, either restarting the tracking of 010 or failing immediately, because we encountered another 010. To minimize the right side of the accepting 010, we can identify $\overline{0}$ with 010.