1.36)

First, realize what $B_n$ means.

\[
B_1 = \{ a, aa, aaa, aaaaa, aaaaaa, \ldots \} \\
B_2 = \{ aa, aaaa, aaaaaa, aaaaaaaaa, \ldots \} \\
B_3 = \{ aaa, aaaaaa, aaaaaaaaa, \ldots \}
\]

Now it is clear what $B_n$ means for $n > 3$.

Now it should be clear how to construct a DFA that accepts $B_n$ for $n > 3$.

1.39)

Let $A_k$ be the set of words of length at least $k - 1$. It’s clear that $A_k$ has $k$ equivalence classes, corresponding to words of length $0, 1, 2, \ldots, k - 2$ and $k - 1$ or more. Thus $A_k$ requires a DFA with $k$ states.
1.41)

Proof by Construction:

Since A is a regular language, there is some DFA that recognizes A. Since B is a regular language, there is some DFA that recognizes B. Construct an NFA C that accepts the perfect shuffle of A and B as follows:
1. Let the start state of DFA A be the start state of NFA C and add a new state, the accept state of NFA C.
2. Add an ε-transition from every accept state of DFA A to the start state of DFA B.
3. Add an ε-transition from every accept state of DFA B to the accept state of NFA C.
4. Add an ε-transition from the accept state of NFA C to the start state of DFA A.

1.46 a)

Assume L = \{ 0^m1^n0^m | m, n \geq 0 \} is regular. Let p be the pumping length given by the pumping lemma. The string s = 0^p1^p0^p \in L, and |s| \geq p. Thus the pumping lemma implies that s can be divided as xyz with x = 0^a, y = 0^b, z = 0^c1^p, where b \geq 1 and a + b + c = p. However, the string s' = xy^0z = 0^{a+c}0^p \notin L, since a + c < p. That contradicts the pumping lemma.

Note: This proof does not choose strings x, y, and z; the notation “x = 0^a, y = 0^b, z = 0^c1^p, where b \geq 1 and a + b + c = p” covers all possible ways of choosing xyz given our choice of s and the constraints of the pumping lemma.

Remember: You are free to choose any s in the language of length at least p, but then you must argue for any legal, possible way of dividing s into xyz. A reasonable choice of s limits the number of cases that must be considered.

1.46 c)

Assume C = \{ w | w \in \{0,1\}^* is a palindrome \} is regular. Let p be the pumping length given by the pumping lemma. The string s = 0^p1^p0^p \in C, and |s| \geq p. Follow the argument as in part (a). Hence C isn’t regular, so neither is its complement.

Remember: You are free to choose any s in the language of length at least p, but then you must argue for any legal, possible way of dividing s into xyz. A reasonable choice of s limits the number of cases that must be considered.
1.51)

To show that $\equiv_L$ is an equivalence relation we show it is reflexive, symmetric, and transitive. It is reflexive because no string can distinguish $x$ from itself and hence $x \equiv_L x$ for every $x$. It is symmetric because $x$ is distinguishable from $y$ whenever $y$ is distinguishable from $x$. It is transitive because if $w \equiv_L x$ and $x \equiv_L y$, then for each $z$, $wz \in L$ iff $xz \in L$ and $xz \in L$ iff $yz \in L$, hence $wz \in L$ iff $yz \in L$, and so $w \equiv_L y$.

1.55 c)

The minimum pumping length is 4. We cannot pump 001, but we can pump the first symbol of any string of length 4 or more.

1.55 e)

The minimum pumping length is 1. The pumping length cannot be 0, as in part (b). Any string in $(01)^*$ of length 1 or more contains 01 and hence can be pumped by dividing it so that $x = \varepsilon$, $u = 01$, and $z$ is the rest.

Note: I also accepted 2 as a correct answer for this one.

1.55 g)

The minimum pumping length is 3. We cannot pump 00, but we can pump the 1 in 100 or 010 or 001.

1.60)

![Diagram](image)
Now it should be clear how to construct an NFA with $k + 1$ states that recognizes $C_k$ for any $k \geq 1$.

Formally, the $\delta$ function specifies a transition from $q_0$ to $q_0$ transition to recognize the $\Sigma^*$ part of $C_k = \Sigma^* a \Sigma^{k-1}$, a transition from $q_0$ to $q_1$ to recognize the $a$ in $C_k = \Sigma^* a \Sigma^{k-1}$, and transitions from $q_i$ to $q_{i+1}$ for $i = 1 \ldots k-1$ to recognize the $\Sigma^{k-1}$ part of $C_k = \Sigma^* a \Sigma^{k-1}$.

1.61)

From Theorem 1.39:
If $k$ is the number of states of the NFA, the DFA simulating the NFA will have $2^k$ states.

Since it is clear from 1.60 that $k + 1$ states are necessary to recognize the $a$ and $\Sigma^{k-1}$ part of $C_k = \Sigma^* a \Sigma^{k-1}$, a DFA that recognizes $C_k$ will have $2^{k+1}$ states.