1. Problem 11(d) on page 38 of the fifth edition, problem 14(d) on page 29 of the sixth edition.

If $M = (Q, \Sigma, \delta, q_0, F)$ is an NFA, we can transform $M$ to a regular grammar $G$ which generates $L(M)$, as follows. There is a construction which uses $M$ to define a regular grammar for $L$. There is one variable of $G$ for each state of $M$, and there is one production of $G$ for each transition (arc) of $M$ and one production for each final state of $M$.

(a) Let $Q = \{q_0, q_1, \ldots, q_k\}$. The variables of $G$ will be $A_0, A_1, \ldots, A_k$, and $A_0$ is the start symbol.
(b) If $M$ has a transition $q_i \xrightarrow{a} q_j$ for $a \in \Sigma$, then $G$ has a production $A_i \rightarrow aA_j$.
(c) If $M$ has a transition $q_i \xrightarrow{\lambda} q_j$, then $G$ has a production $A_i \rightarrow A_j$.
(d) If $q_i \in F$, then $G$ has a production $A_i \rightarrow \lambda$.

The language consisting of all strings over $\Sigma = \{a, b\}$ with at least 3 $a$'s is accepted by the following DFA (which is, of course, also an NFA):

```
q_0 ----> a ----> q_1 ----> a ----> q_2 ----> a ----> q_3
    \   \   \   \   \   \  \
     b   b   b   b   a,b
```

The grammar $G$ obtained from $M$ by the transformation given in problem 1 has the productions:

$A_0 \rightarrow bA_0|aA_1$
$A_1 \rightarrow bA_1|aA_2$
$A_2 \rightarrow bA_2|aA_3$
$A_3 \rightarrow aA_3|bA_3|\lambda$

2. Problem 14(a) on page 39 of the fifth edition,

$S \rightarrow aSb$
$S \rightarrow Tb$
$T \rightarrow Tb$
$T \rightarrow \lambda$
3. problem 17(h) on page 29 of the sixth edition. We introduce a third variable, \( S' \), which plays the same role as \( S \) does in the previous problem. The new start symbol \( S \) then generates arbitrarily many copies of \( S' \), each of which generates a member of \( L_1 \).

\[
\begin{align*}
S & \rightarrow S'S \\
S & \rightarrow \lambda \\
S' & \rightarrow aS'b \\
S' & \rightarrow Tb \\
T & \rightarrow Tb \\
T & \rightarrow \lambda
\end{align*}
\]


Instead of showing all 26 letters and ten digits, we simplify the figure by simply writing "digit" instead of 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, similarly for letters.

We use the transformation given in problem 1, and we introduce two variables, \( L \) for letter and \( D \) for digit. Our grammar has variables \( A_0, A_1, A_2, A_3, A_4, L, D \), and the start symbol is \( A_0 \).

\[
\begin{align*}
A_0 & \rightarrow LA_1 \\
A_1 & \rightarrow DA_2 | LA_1 \\
A_2 & \rightarrow DA_4 | LA_2 | \lambda \\
A_3 & \rightarrow DA_4 | LA_3 | \lambda \\
A_4 & \rightarrow LA_4 | \lambda \\
L & \rightarrow a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z \\
D & \rightarrow 0|1|2|3|4|5|6|7|8|9
\end{align*}
\]

5. Problem 2(a) on page 56 of the fifth edition, problem 4(a) on page 48 of the sixth edition.

There is a dead state, not shown in the figure.
6. Problem 8(b), 9(a), 9(c) on page 56 of the fifth edition, 8(b), 11(a), 11(c) on page 49 of the sixth edition.


Figure 2.4 shows a DFA with four states, one of which is a dead state. If we apply the algorithm to find the minimal equivalent DFA, we will still have four states. Thus, the answer is no.


This is the NFA shown in Figure 2-8. Let’s call it \( M \).

We now apply the algorithm to convert an NFA into an equivalent DFA. Each state of the DFA is a subset of the set of states of \( M \). Since \( M \) has 6 states, there are \( 2^6 = 64 \) such subsets altogether. Fortunately, most of those states are useless, and thus we don’t need to include them in our figure. The only ones we show are \( \{ q_0 \} \), \( \{ q_1, q_4 \} \), \( \{ q_2, q_5 \} \), \( \{ q_3, q_4 \} \), \( \{ q_5 \} \), \( \{ q_1 \} \). In the figure, we abbreviate these names as 0, 14, 25, 34, 5, and 4.


The safest way to solve this is to compute a DFA equivalent to the NFA. Using the same notation as in the previous problem, we get three states, which we call 0, 02, and 12. Of these, 0 and 02 are equivalent, hence the NFA in the figure is equivalent to a DFA with only two states:

We can now see that the strings 01001 and 000 are the only ones accepted.