1.4

a. \( L_1 = \{ w \mid w \text{ has at least three } a's \} \)

\[
\begin{array}{c}
\text{a} \\
\text{b} \\
\text{a} \\
\text{b} \\
\text{a} \\
\text{a} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{a} \\
\text{a} \\
\text{b} \\
\text{b} \\
\end{array}
\]

\( L_2 = \{ w \mid w \text{ has at least two } b's \} \)

\[
\begin{array}{c}
\text{a} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{a} \\
\text{b} \\
\text{a} \\
\text{b} \\
\text{b} \\
\end{array}
\]

Combining them using the intersection construction gives the DFA:

\[
\begin{array}{c}
\text{a} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{a} \\
\text{a} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{b} \\
\text{a} \\
\text{a} \\
\text{b} \\
\text{b} \\
\end{array}
\]

c. \( L_3 = \{ w \mid w \text{ has an even number of } a's \} \)

\[
\begin{array}{c}
\text{b} \\
\text{a} \\
\text{b} \\
\text{b} \\
\end{array}
\]
\[ L_2 = \{w \mid w \text{ has one or two } b's \} \]

Combining them using the intersection construction gives the DFA:

\[ e. \ L_1 = \{w \mid w \text{ starts with an } a \} \]

\[ L_2 = \{w \mid w \text{ has at most one } b \} \]
Combining them using the intersection construction gives the DFA:

\[ L_1 = \{ w \mid w \text{ has an odd number of } a's \} \]

\[ L_2 = \{ w \mid w \text{ ends with a } b \} \]
Combining them using the intersection construction gives the DFA:

\[
L_{w_1} = \{ w \mid w \text{ has even length} \}
\]

\[
L_{w_2} = \{ w \mid w \text{ has an odd number of } a\text{'s} \}
\]
Combining them using the intersection construction gives the DFA:

1.6
c.

e.
1.

1.7

c.

d.

g.
1.14

a. Proof:

As for DFA, at each state, it will consider all kinds of input, and make exactly one kind of transition according to the given input. And if language $B$ is recognized by DFA $M$, it means that any string belongs to $B$ will begin from the start state of $M$ and reach the accept state of $M$ at the end. Hence, if we swap the accept states and nonaccept states of $M$, strings of language $B$ will no longer be acceptable for this new DFA. Instead, the new DFA can accept any regular language other than $B$, which implies that the new DFA can recognize the complement of $B$.

Because the new DFA is also a DFA, so the language accepted by this new DFA is also regular language, which means that the class of regular languages is closed under complement.

b.

As for NFA, at each state, for a certain kind of input, it may have different ways of transition. So, swapping the accept and nonaccept states doesn’t necessarily yield a new NFA that recognizes the complement of $C$.

Here is an example. Consider NFA $N_2$ in textbook, page 51. Originally, it can accept all strings over \{0,1\} containing a 1 in the third position from the end, which means it can accept string 000100. After swapping the accept states and nonaccept states, it can still accept the same string, 000100. Clearly, in this case, the new NFA cannot recognize the complement of $C$.

Class of languages recognized by NFAs is closed under complement. Because, every NFA has an equivalent DFA, so for any language that can be accepted by a NFA, we can construct an equivalent DFA, then swap the accept states and nonaccept states of the constructed DFA, we will have a new DFA that recognize the complement of original language.
1.17

a.

b.
The answer to this problem could be different.

a) Member: ab; aabb
   Not Member: aba; abba

b) Member: abab; ab
   Not Member: aaba; ba

c) Member: a; b
   Not Member: ab; bba

d) Member: aaa; ε
   Not Member: ab; ba

e) Member: aba; aaba
   Not Member: bbb; aaa

f) Member: aba; bab
   Not Member: aa; bb

g) Member: b; ab
   Not Member: ba; bba

h) Member: a; ba
   Not Member: aaa; bbb