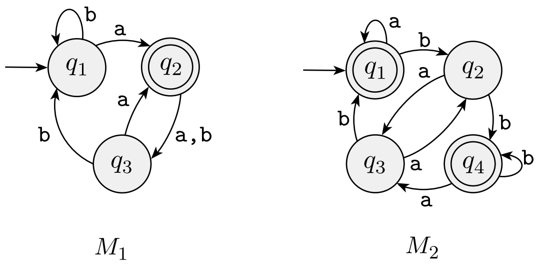
EXERCISES

[A]**1.1** The following are the state diagrams of two DFAs, M_1 and M_2. Answer the following questions about each of these machines.



a. What is the start state?
b. What is the set of accept states?
c. What sequence of states does the machine go through on input aabb?
d. Does the machine accept the string aabb?
e. Does the machine accept the string ε?

[A]**1.2** Give the formal description of the machines M_1 and M_2 pictured in Exercise 1.1.

1.3 The formal description of a DFA M is $(\{q_1, q_2, q_3, q_4, q_5\}, \{\texttt{u}, \texttt{d}\}, \delta, q_3, \{q_3\})$, where δ is given by the following table. Give the state diagram of this machine.

	u	d
q_1	q_1	q_2
q_2	q_1	q_3
q_3	q_2	q_4
q_4	q_3	q_5
q_5	q_4	q_5

1.4 Each of the following languages is the intersection of two simpler languages. In each part, construct DFAs for the simpler languages, then combine them using the construction discussed in footnote 3 (page 46) to give the state diagram of a DFA for the language given. In all parts, $\Sigma = \{\texttt{a}, \texttt{b}\}$.

a. $\{w|\ w$ has at least three a's and at least two b's$\}$
[A]**b.** $\{w|\ w$ has exactly two a's and at least two b's$\}$
c. $\{w|\ w$ has an even number of a's and one or two b's$\}$
[A]**d.** $\{w|\ w$ has an even number of a's and each a is followed by at least one b$\}$
e. $\{w|\ w$ starts with an a and has at most one b$\}$
f. $\{w|\ w$ has an odd number of a's and ends with a b$\}$
g. $\{w|\ w$ has even length and an odd number of a's$\}$

1.5 Each of the following languages is the complement of a simpler language. In each part, construct a DFA for the simpler language, then use it to give the state diagram of a DFA for the language given. In all parts, $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

- ${}^{\mathsf{A}}$**a.** $\{w \mid w$ does not contain the substring $\mathtt{ab}\}$
- ${}^{\mathsf{A}}$**b.** $\{w \mid w$ does not contain the substring $\mathtt{baba}\}$
- **c.** $\{w \mid w$ contains neither the substrings $\mathtt{ab}$ nor $\mathtt{ba}\}$
- **d.** $\{w \mid w$ is any string not in $\mathtt{a}^*\mathtt{b}^*\}$
- **e.** $\{w \mid w$ is any string not in $(\mathtt{ab}^+)^*\}$
- **f.** $\{w \mid w$ is any string not in $\mathtt{a}^* \cup \mathtt{b}^*\}$
- **g.** $\{w \mid w$ is any string that doesn't contain exactly two $\mathtt{a}$'s$\}$
- **h.** $\{w \mid w$ is any string except $\mathtt{a}$ and $\mathtt{b}\}$

1.6 Give state diagrams of DFAs recognizing the following languages. In all parts, the alphabet is $\{\mathtt{0},\mathtt{1}\}$.

- **a.** $\{w \mid w$ begins with a $\mathtt{1}$ and ends with a $\mathtt{0}\}$
- **b.** $\{w \mid w$ contains at least three $\mathtt{1}$s$\}$
- **c.** $\{w \mid w$ contains the substring $\mathtt{0101}$ (i.e., $w = x\mathtt{0101}y$ for some x and y)$\}$
- **d.** $\{w \mid w$ has length at least 3 and its third symbol is a $\mathtt{0}\}$
- **e.** $\{w \mid w$ starts with $\mathtt{0}$ and has odd length, or starts with $\mathtt{1}$ and has even length$\}$
- **f.** $\{w \mid w$ doesn't contain the substring $\mathtt{110}\}$
- **g.** $\{w \mid$ the length of w is at most $5\}$
- **h.** $\{w \mid w$ is any string except $\mathtt{11}$ and $\mathtt{111}\}$
- **i.** $\{w \mid$ every odd position of w is a $\mathtt{1}\}$
- **j.** $\{w \mid w$ contains at least two $\mathtt{0}$s and at most one $\mathtt{1}\}$
- **k.** $\{\varepsilon, \mathtt{0}\}$
- **l.** $\{w \mid w$ contains an even number of $\mathtt{0}$s, or contains exactly two $\mathtt{1}$s$\}$
- **m.** The empty set
- **n.** All strings except the empty string

1.7 Give state diagrams of NFAs with the specified number of states recognizing each of the following languages. In all parts, the alphabet is $\{\mathtt{0},\mathtt{1}\}$.

- ${}^{\mathsf{A}}$**a.** The language $\{w \mid w$ ends with $\mathtt{00}\}$ with three states
- **b.** The language of Exercise 1.6c with five states
- **c.** The language of Exercise 1.6l with six states
- **d.** The language $\{\mathtt{0}\}$ with two states
- **e.** The language $\mathtt{0}^*\mathtt{1}^*\mathtt{0}^+$ with three states
- ${}^{\mathsf{A}}$**f.** The language $\mathtt{1}^*(\mathtt{001}^+)^*$ with three states
- **g.** The language $\{\varepsilon\}$ with one state
- **h.** The language $\mathtt{0}^*$ with one state

1.8 Use the construction in the proof of Theorem 1.45 to give the state diagrams of NFAs recognizing the union of the languages described in

- **a.** Exercises 1.6a and 1.6b.
- **b.** Exercises 1.6c and 1.6f.

1.9 Use the construction in the proof of Theorem 1.47 to give the state diagrams of NFAs recognizing the concatenation of the languages described in

a. Exercises 1.6g and 1.6i.
b. Exercises 1.6b and 1.6m.

1.10 Use the construction in the proof of Theorem 1.49 to give the state diagrams of NFAs recognizing the star of the languages described in

a. Exercise 1.6b.
b. Exercise 1.6j.
c. Exercise 1.6m.

^A**1.11** Prove that every NFA can be converted to an equivalent one that has a single accept state.

1.12 Let $D = \{w|\ w$ contains an even number of `a`'s and an odd number of `b`'s and does not contain the substring `ab`$\}$. Give a DFA with five states that recognizes D and a regular expression that generates D. (Suggestion: Describe D more simply.)

1.13 Let F be the language of all strings over $\{$`0`,`1`$\}$ that do not contain a pair of `1`s that are separated by an odd number of symbols. Give the state diagram of a DFA with five states that recognizes F. (You may find it helpful first to find a 4-state NFA for the complement of F.)

1.14 **a.** Show that if M is a DFA that recognizes language B, swapping the accept and nonaccept states in M yields a new DFA recognizing the complement of B. Conclude that the class of regular languages is closed under complement.
b. Show by giving an example that if M is an NFA that recognizes language C, swapping the accept and nonaccept states in M doesn't necessarily yield a new NFA that recognizes the complement of C. Is the class of languages recognized by NFAs closed under complement? Explain your answer.

1.15 Give a counterexample to show that the following construction fails to prove Theorem 1.49, the closure of the class of regular languages under the star operation.[7] Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize A_1. Construct $N = (Q_1, \Sigma, \delta, q_1, F)$ as follows. N is supposed to recognize A_1^*.

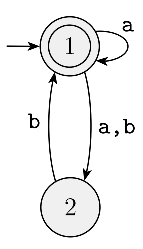
a. The states of N are the states of N_1.
b. The start state of N is the same as the start state of N_1.
c. $F = \{q_1\} \cup F_1$.
The accept states F are the old accept states plus its start state.
d. Define δ so that for any $q \in Q_1$ and any $a \in \Sigma_\varepsilon$,

$$\delta(q,a) = \begin{cases} \delta_1(q,a) & q \notin F_1 \text{ or } a \neq \varepsilon \\ \delta_1(q,a) \cup \{q_1\} & q \in F_1 \text{ and } a = \varepsilon. \end{cases}$$

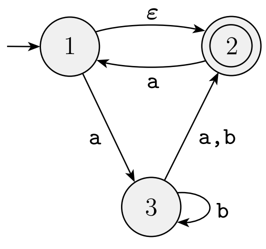
(Suggestion: Show this construction graphically, as in Figure 1.50.)

[7]In other words, you must present a finite automaton, N_1, for which the constructed automaton N does not recognize the star of N_1's language.

1.16 Use the construction given in Theorem 1.39 to convert the following two nondeterministic finite automata to equivalent deterministic finite automata.



(a)



(b)

1.17 **a.** Give an NFA recognizing the language $(01 \cup 001 \cup 010)^*$.
b. Convert this NFA to an equivalent DFA. Give only the portion of the DFA that is reachable from the start state.

1.18 Give regular expressions generating the languages of Exercise 1.6.

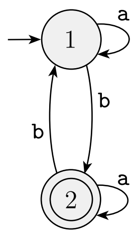
1.19 Use the procedure described in Lemma 1.55 to convert the following regular expressions to nondeterministic finite automata.

a. $(0 \cup 1)^*000(0 \cup 1)^*$
b. $(((00)^*(11)) \cup 01)^*$
c. $\emptyset^*$

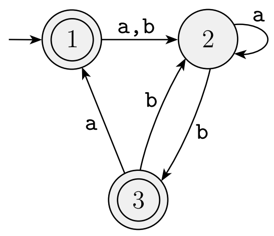
1.20 For each of the following languages, give two strings that are members and two strings that are *not* members—a total of four strings for each part. Assume the alphabet $\Sigma = \{a,b\}$ in all parts.

a. a^*b^*
b. $a(ba)^*b$
c. $a^* \cup b^*$
d. $(aaa)^*$
e. $\Sigma^*a\Sigma^*b\Sigma^*a\Sigma^*$
f. $aba \cup bab$
g. $(\varepsilon \cup a)b$
h. $(a \cup ba \cup bb)\Sigma^*$

1.21 Use the procedure described in Lemma 1.60 to convert the following finite automata to regular expressions.



(a)



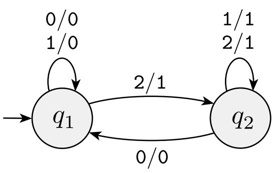
(b)

1.22 In certain programming languages, comments appear between delimiters such as `/#` and `#/`. Let C be the language of all valid delimited comment strings. A member of C must begin with `/#` and end with `#/` but have no intervening `#/`. For simplicity, assume that the alphabet for C is $\Sigma = \{\texttt{a}, \texttt{b}, \texttt{/}, \texttt{\#}\}$.

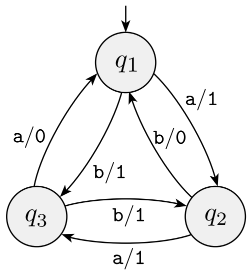
- **a.** Give a DFA that recognizes C.
- **b.** Give a regular expression that generates C.

$^{\mathsf{A}}$**1.23** Let B be any language over the alphabet Σ. Prove that $B = B^+$ iff $BB \subseteq B$.

1.24 A ***finite state transducer*** (FST) is a type of deterministic finite automaton whose output is a string and not just *accept* or *reject*. The following are state diagrams of finite state transducers T_1 and T_2.



T_1



T_2

Each transition of an FST is labeled with two symbols, one designating the input symbol for that transition and the other designating the output symbol. The two symbols are written with a slash, /, separating them. In T_1, the transition from q_1 to q_2 has input symbol `2` and output symbol `1`. Some transitions may have multiple input–output pairs, such as the transition in T_1 from q_1 to itself. When an FST computes on an input string w, it takes the input symbols $w_1 \cdots w_n$ one by one and, starting at the start state, follows the transitions by matching the input labels with the sequence of symbols $w_1 \cdots w_n = w$. Every time it goes along a transition, it outputs the corresponding output symbol. For example, on input `2212011`, machine T_1 enters the sequence of states q_1, q_2, q_2, q_2, q_2, q_1, q_1, q_1 and produces output `1111000`. On input `abbb`, T_2 outputs `1011`. Give the sequence of states entered and the output produced in each of the following parts.

- **a.** T_1 on input `011`
- **b.** T_1 on input `211`
- **c.** T_1 on input `121`
- **d.** T_1 on input `0202`
- **e.** T_2 on input `b`
- **f.** T_2 on input `bbab`
- **g.** T_2 on input `bbbbbb`
- **h.** T_2 on input ε

1.25 Read the informal definition of the finite state transducer given in Exercise 1.24. Give a formal definition of this model, following the pattern in Definition 1.5 (page 35). Assume that an FST has an input alphabet Σ and an output alphabet Γ but not a set of accept states. Include a formal definition of the computation of an FST. (Hint: An FST is a 5-tuple. Its transition function is of the form $\delta\colon Q \times \Sigma \longrightarrow Q \times \Gamma$.)

1.26 Using the solution you gave to Exercise 1.25, give a formal description of the machines T_1 and T_2 depicted in Exercise 1.24.

1.27 Read the informal definition of the finite state transducer given in Exercise 1.24. Give the state diagram of an FST with the following behavior. Its input and output alphabets are $\{\texttt{0},\texttt{1}\}$. Its output string is identical to the input string on the even positions but inverted on the odd positions. For example, on input `0000111` it should output `1010010`.

1.28 Convert the following regular expressions to NFAs using the procedure given in Theorem 1.54. In all parts, $\Sigma = \{\texttt{a}, \texttt{b}\}$.

a. $\texttt{a}(\texttt{abb})^* \cup \texttt{b}$
b. $\texttt{a}^+ \cup (\texttt{ab})^+$
c. $(\texttt{a} \cup \texttt{b}^+)\texttt{a}^+\texttt{b}^+$

1.29 Use the pumping lemma to show that the following languages are not regular.

$^{\mathsf{A}}$**a.** $A_1 = \{\texttt{0}^n\texttt{1}^n\texttt{2}^n \mid n \geq 0\}$
b. $A_2 = \{www \mid w \in \{\texttt{a},\texttt{b}\}^*\}$
$^{\mathsf{A}}$**c.** $A_3 = \{\texttt{a}^{2^n} \mid n \geq 0\}$ (Here, $\texttt{a}^{2^n}$ means a string of 2^n a's.)

1.30 Describe the error in the following "proof" that $\texttt{0}^*\texttt{1}^*$ is not a regular language. (An error must exist because $\texttt{0}^*\texttt{1}^*$ *is* regular.) The proof is by contradiction. Assume that $\texttt{0}^*\texttt{1}^*$ is regular. Let p be the pumping length for $\texttt{0}^*\texttt{1}^*$ given by the pumping lemma. Choose s to be the string $\texttt{0}^p\texttt{1}^p$. You know that s is a member of $\texttt{0}^*\texttt{1}^*$, but Example 1.73 shows that s cannot be pumped. Thus you have a contradiction. So $\texttt{0}^*\texttt{1}^*$ is not regular.

PROBLEMS

1.31 For any string $w = w_1w_2\cdots w_n$, the ***reverse*** of w, written $w^{\mathcal{R}}$, is the string w in reverse order, $w_n \cdots w_2w_1$. For any language A, let $A^{\mathcal{R}} = \{w^{\mathcal{R}} \mid w \in A\}$.
Show that if A is regular, so is $A^{\mathcal{R}}$.

1.32 Let

$$\Sigma_3 = \left\{ \begin{bmatrix}0\\0\\0\end{bmatrix}, \begin{bmatrix}0\\0\\1\end{bmatrix}, \begin{bmatrix}0\\1\\0\end{bmatrix}, \ldots, \begin{bmatrix}1\\1\\1\end{bmatrix} \right\}.$$

Σ_3 contains all size 3 columns of 0s and 1s. A string of symbols in Σ_3 gives three rows of 0s and 1s. Consider each row to be a binary number and let

$$B = \{w \in \Sigma_3^* \mid \text{the bottom row of } w \text{ is the sum of the top two rows}\}.$$

For example,

$$\begin{bmatrix}0\\0\\1\end{bmatrix}\begin{bmatrix}1\\0\\0\end{bmatrix}\begin{bmatrix}1\\1\\0\end{bmatrix} \in B, \quad \text{but} \quad \begin{bmatrix}0\\0\\1\end{bmatrix}\begin{bmatrix}1\\0\\1\end{bmatrix} \notin B.$$

Show that B is regular. (Hint: Working with $B^{\mathcal{R}}$ is easier. You may assume the result claimed in Problem 1.31.)

1.33 Let
$$\Sigma_2 = \left\{ \begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}1\\0\end{bmatrix}, \begin{bmatrix}1\\1\end{bmatrix} \right\}.$$
Here, Σ_2 contains all columns of 0s and 1s of height two. A string of symbols in Σ_2 gives two rows of 0s and 1s. Consider each row to be a binary number and let

$$C = \{w \in \Sigma_2^* |\ \text{the bottom row of } w \text{ is three times the top row}\}.$$

For example, $\begin{bmatrix}0\\0\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}0\\0\end{bmatrix} \in C$, but $\begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix} \notin C$. Show that C is regular. (You may assume the result claimed in Problem 1.31.)

1.34 Let Σ_2 be the same as in Problem 1.33. Consider each row to be a binary number and let

$$D = \{w \in \Sigma_2^* |\ \text{the top row of } w \text{ is a larger number than is the bottom row}\}.$$

For example, $\begin{bmatrix}0\\0\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}0\\0\end{bmatrix} \in D$, but $\begin{bmatrix}0\\0\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}0\\0\end{bmatrix} \notin D$. Show that D is regular.

1.35 Let Σ_2 be the same as in Problem 1.33. Consider the top and bottom rows to be strings of 0s and 1s, and let

$$E = \{w \in \Sigma_2^* |\ \text{the bottom row of } w \text{ is the reverse of the top row of } w\}.$$

Show that E is not regular.

1.36 Let $B_n = \{\mathtt{a}^k |\ k \text{ is a multiple of } n\}$. Show that for each $n \geq 1$, the language B_n is regular.

1.37 Let $C_n = \{x |\ x \text{ is a binary number that is a multiple of } n\}$. Show that for each $n \geq 1$, the language C_n is regular.

1.38 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, in contrast, that an ordinary NFA accepts a string if *some* state among these possible states is an accept state. Prove that all-NFAs recognize the class of regular languages.

1.39 The construction in Theorem 1.54 shows that every GNFA is equivalent to a GNFA with only two states. We can show that an opposite phenomenon occurs for DFAs. Prove that for every $k > 1$, a language $A_k \subseteq \{0,1\}^*$ exists that is recognized by a DFA with k states but not by one with only $k-1$ states.

1.40 Recall that string x is a ***prefix*** of string y if a string z exists where $xz = y$, and that x is a ***proper prefix*** of y if in addition $x \neq y$. In each of the following parts, we define an operation on a language A. Show that the class of regular languages is closed under that operation.

$^{\mathsf{A}}$**a.** $NOPREFIX(A) = \{w \in A |\ \text{no proper prefix of } w \text{ is a member of } A\}$.

b. $NOEXTEND(A) = \{w \in A |\ w \text{ is not the proper prefix of any string in } A\}$.

1.41 For languages A and B, let the ***perfect shuffle*** of A and B be the language

$$\{w |\ w = a_1 b_1 \cdots a_k b_k,\ \text{where } a_1 \cdots a_k \in A \text{ and } b_1 \cdots b_k \in B,\ \text{each } a_i, b_i \in \Sigma\}.$$

Show that the class of regular languages is closed under perfect shuffle.

1.42 For languages A and B, let the ***shuffle*** of A and B be the language

$$\{w |\ w = a_1 b_1 \cdots a_k b_k,\ \text{where } a_1 \cdots a_k \in A \text{ and } b_1 \cdots b_k \in B,\ \text{each } a_i, b_i \in \Sigma^*\}.$$

Show that the class of regular languages is closed under shuffle.

1.43 Let A be any language. Define *DROP-OUT*(A) to be the language containing all strings that can be obtained by removing one symbol from a string in A. Thus, *DROP-OUT*$(A) = \{xz \mid xyz \in A \text{ where } x, z \in \Sigma^*, y \in \Sigma\}$. Show that the class of regular languages is closed under the *DROP-OUT* operation. Give both a proof by picture and a more formal proof by construction as in Theorem 1.47.

^{A}**1.44** Let B and C be languages over $\Sigma = \{\texttt{0}, \texttt{1}\}$. Define

$$B \overset{1}{\leftarrow} C = \{w \in B \mid \text{ for some } y \in C, \text{ strings } w \text{ and } y \text{ contain equal numbers of } \texttt{1}\text{s}\}.$$

Show that the class of regular languages is closed under the $\overset{1}{\leftarrow}$ operation.

***1.45** Let $A/B = \{w \mid wx \in A \text{ for some } x \in B\}$. Show that if A is regular and B is any language, then A/B is regular.

1.46 Prove that the following languages are not regular. You may use the pumping lemma and the closure of the class of regular languages under union, intersection, and complement.

a. $\{\texttt{0}^n\texttt{1}^m\texttt{0}^n \mid m, n \geq 0\}$
^{A}**b.** $\{\texttt{0}^m\texttt{1}^n \mid m \neq n\}$
c. $\{w \mid w \in \{\texttt{0},\texttt{1}\}^* \text{ is not a palindrome}\}$[8]
***d.** $\{wtw \mid w, t \in \{\texttt{0},\texttt{1}\}^+\}$

1.47 Let $\Sigma = \{\texttt{1}, \texttt{\#}\}$ and let

$$Y = \{w \mid w = x_1\texttt{\#}x_2\texttt{\#}\cdots\texttt{\#}x_k \text{ for } k \geq 0, \text{ each } x_i \in \texttt{1}^*, \text{ and } x_i \neq x_j \text{ for } i \neq j\}.$$

Prove that Y is not regular.

1.48 Let $\Sigma = \{\texttt{0},\texttt{1}\}$ and let

$$D = \{w \mid w \text{ contains an equal number of occurrences of the substrings } \texttt{01} \text{ and } \texttt{10}\}.$$

Thus $\texttt{101} \in D$ because $\texttt{101}$ contains a single $\texttt{01}$ and a single $\texttt{10}$, but $\texttt{1010} \notin D$ because $\texttt{1010}$ contains two $\texttt{10}$s and one $\texttt{01}$. Show that D is a regular language.

1.49 **a.** Let $B = \{\texttt{1}^k y \mid y \in \{\texttt{0}, \texttt{1}\}^* \text{ and } y \text{ contains at least } k \text{ } \texttt{1}\text{s, for } k \geq 1\}$. Show that B is a regular language.
b. Let $C = \{\texttt{1}^k y \mid y \in \{\texttt{0}, \texttt{1}\}^* \text{ and } y \text{ contains at most } k \text{ } \texttt{1}\text{s, for } k \geq 1\}$. Show that C isn't a regular language.

^{A}**1.50** Read the informal definition of the finite state transducer given in Exercise 1.24. Prove that no FST can output $w^{\mathcal{R}}$ for every input w if the input and output alphabets are $\{\texttt{0},\texttt{1}\}$.

1.51 Let x and y be strings and let L be any language. We say that x and y are ***distinguishable by L*** if some string z exists whereby exactly one of the strings xz and yz is a member of L; otherwise, for every string z, we have $xz \in L$ whenever $yz \in L$ and we say that x and y are ***indistinguishable by L***. If x and y are indistinguishable by L, we write $x \equiv_L y$. Show that $\equiv_L$ is an equivalence relation.

[8]A ***palindrome*** is a string that reads the same forward and backward.

^{A*}**1.52** **Myhill–Nerode theorem.** Refer to Problem 1.51. Let L be a language and let X be a set of strings. Say that X is ***pairwise distinguishable by L*** if every two distinct strings in X are distinguishable by L. Define the ***index of L*** to be the maximum number of elements in any set that is pairwise distinguishable by L. The index of L may be finite or infinite.

- **a.** Show that if L is recognized by a DFA with k states, L has index at most k.
- **b.** Show that if the index of L is a finite number k, it is recognized by a DFA with k states.
- **c.** Conclude that L is regular iff it has finite index. Moreover, its index is the size of the smallest DFA recognizing it.

1.53 Let $\Sigma = \{\texttt{0}, \texttt{1}, \texttt{+}, \texttt{=}\}$ and

$$ADD = \{x\texttt{=}y\texttt{+}z \mid x, y, z \text{ are binary integers, and } x \text{ is the sum of } y \text{ and } z\}.$$

Show that ADD is not regular.

1.54 Consider the language $F = \{\texttt{a}^i\texttt{b}^j\texttt{c}^k \mid i, j, k \geq 0 \text{ and if } i = 1 \text{ then } j = k\}$.

- **a.** Show that F is not regular.
- **b.** Show that F acts like a regular language in the pumping lemma. In other words, give a pumping length p and demonstrate that F satisfies the three conditions of the pumping lemma for this value of p.
- **c.** Explain why parts (a) and (b) do not contradict the pumping lemma.

1.55 The pumping lemma says that every regular language has a pumping length p, such that every string in the language can be pumped if it has length p or more. If p is a pumping length for language A, so is any length $p' \geq p$. The ***minimum pumping length*** for A is the smallest p that is a pumping length for A. For example, if $A = \texttt{01}^*$, the minimum pumping length is 2. The reason is that the string $s = \texttt{0}$ is in A and has length 1 yet s cannot be pumped; but any string in A of length 2 or more contains a 1 and hence can be pumped by dividing it so that $x = \texttt{0}$, $y = \texttt{1}$, and z is the rest. For each of the following languages, give the minimum pumping length and justify your answer.

- ^{A}**a.** $\texttt{0001}^*$
- ^{A}**b.** $\texttt{0}^*\texttt{1}^*$
- **c.** $\texttt{001} \cup \texttt{0}^*\texttt{1}^*$
- ^{A}**d.** $\texttt{0}^*\texttt{1}^+\texttt{0}^+\texttt{1}^* \cup \texttt{10}^*\texttt{1}$
- **e.** $(\texttt{01})^*$
- **f.** ε
- **g.** $\texttt{1}^*\texttt{01}^*\texttt{01}^*$
- **h.** $\texttt{10}(\texttt{11}^*\texttt{0})^*\texttt{0}$
- **i.** $\texttt{1011}$
- **j.** Σ^*

***1.56** If A is a set of natural numbers and k is a natural number greater than 1, let

$$B_k(A) = \{w \mid w \text{ is the representation in base } k \text{ of some number in } A\}.$$

Here, we do not allow leading 0s in the representation of a number. For example, $B_2(\{3, 5\}) = \{\texttt{11}, \texttt{101}\}$ and $B_3(\{3, 5\}) = \{\texttt{10}, \texttt{12}\}$. Give an example of a set A for which $B_2(A)$ is regular but $B_3(A)$ is not regular. Prove that your example works.

*1.57 If A is any language, let $A_{\frac{1}{2}-}$ be the set of all first halves of strings in A so that

$$A_{\frac{1}{2}-} = \{x \mid \text{ for some } y,\ |x| = |y| \text{ and } xy \in A\}.$$

Show that if A is regular, then so is $A_{\frac{1}{2}-}$.

*1.58 If A is any language, let $A_{\frac{1}{3}-\frac{1}{3}}$ be the set of all strings in A with their middle thirds removed so that

$$A_{\frac{1}{3}-\frac{1}{3}} = \{xz \mid \text{ for some } y,\ |x| = |y| = |z| \text{ and } xyz \in A\}.$$

Show that if A is regular, then $A_{\frac{1}{3}-\frac{1}{3}}$ is not necessarily regular.

*1.59 Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA and let h be a state of M called its "home". A ***synchronizing sequence*** for M and h is a string $s \in \Sigma^*$ where $\delta(q, s) = h$ for every $q \in Q$. (Here we have extended δ to strings, so that $\delta(q, s)$ equals the state where M ends up when M starts at state q and reads input s.) Say that M is ***synchronizable*** if it has a synchronizing sequence for some state h. Prove that if M is a k-state synchronizable DFA, then it has a synchronizing sequence of length at most k^3. Can you improve upon this bound?

1.60 Let $\Sigma = \{\mathtt{a}, \mathtt{b}\}$. For each $k \geq 1$, let C_k be the language consisting of all strings that contain an $\mathtt{a}$ exactly k places from the right-hand end. Thus $C_k = \Sigma^*\mathtt{a}\Sigma^{k-1}$. Describe an NFA with $k + 1$ states that recognizes C_k in terms of both a state diagram and a formal description.

1.61 Consider the languages C_k defined in Problem 1.60. Prove that for each k, no DFA can recognize C_k with fewer than 2^k states.

1.62 Let $\Sigma = \{\mathtt{a}, \mathtt{b}\}$. For each $k \geq 1$, let D_k be the language consisting of all strings that have at least one $\mathtt{a}$ among the last k symbols. Thus $D_k = \Sigma^*\mathtt{a}(\Sigma \cup \varepsilon)^{k-1}$. Describe a DFA with at most $k+1$ states that recognizes D_k in terms of both a state diagram and a formal description.

*1.63

a. Let A be an infinite regular language. Prove that A can be split into two infinite disjoint regular subsets.

b. Let B and D be two languages. Write $B \Subset D$ if $B \subseteq D$ and D contains infinitely many strings that are not in B. Show that if B and D are two regular languages where $B \Subset D$, then we can find a regular language C where $B \Subset C \Subset D$.

1.64 Let N be an NFA with k states that recognizes some language A.

a. Show that if A is nonempty, A contains some string of length at most k.

b. Show, by giving an example, that part (a) is not necessarily true if you replace both A's by $\overline{A}$.

c. Show that if $\overline{A}$ is nonempty, $\overline{A}$ contains some string of length at most 2^k.

d. Show that the bound given in part (c) is nearly tight; that is, for each k, demonstrate an NFA recognizing a language A_k where $\overline{A_k}$ is nonempty and where $\overline{A_k}$'s shortest member strings are of length exponential in k. Come as close to the bound in (c) as you can.

***1.65** Prove that for each $n > 0$, a language B_n exists where

a. B_n is recognizable by an NFA that has n states, and

b. if $B_n = A_1 \cup \cdots \cup A_k$, for regular languages A_i, then at least one of the A_i requires a DFA with exponentially many states.

1.66 A ***homomorphism*** is a function $f\colon \Sigma \longrightarrow \Gamma^*$ from one alphabet to strings over another alphabet. We can extend f to operate on strings by defining $f(w) = f(w_1)f(w_2)\cdots f(w_n)$, where $w = w_1w_2\cdots w_n$ and each $w_i \in \Sigma$. We further extend f to operate on languages by defining $f(A) = \{f(w)|\ w \in A\}$, for any language A.

a. Show, by giving a formal construction, that the class of regular languages is closed under homomorphism. In other words, given a DFA M that recognizes B and a homomorphism f, construct a finite automaton M' that recognizes $f(B)$. Consider the machine M' that you constructed. Is it a DFA in every case?

b. Show, by giving an example, that the class of non-regular languages is not closed under homomorphism.

***1.67** Let the ***rotational closure*** of language A be $RC(A) = \{yx|\ xy \in A\}$.

a. Show that for any language A, we have $RC(A) = RC(RC(A))$.

b. Show that the class of regular languages is closed under rotational closure.

***1.68** In the traditional method for cutting a deck of playing cards, the deck is arbitrarily split two parts, which are exchanged before reassembling the deck. In a more complex cut, called Scarne's cut, the deck is broken into three parts and the middle part in placed first in the reassembly. We'll take Scarne's cut as the inspiration for an operation on languages. For a language A, let $CUT(A) = \{yxz|\ xyz \in A\}$.

a. Exhibit a language B for which $CUT(B) \neq CUT(CUT(B))$.

b. Show that the class of regular languages is closed under CUT.

1.69 Let $\Sigma = \{\texttt{0},\texttt{1}\}$. Let $WW_k = \{ww|\ w \in \Sigma^*$ and w is of length $k\}$.

a. Show that for each k, no DFA can recognize WW_k with fewer than 2^k states.

b. Describe a much smaller NFA for $\overline{WW_k}$, the complement of WW_k.

1.70 We define the ***avoids*** operation for languages A and B to be

$$A \textit{ avoids } B = \{w|\ w \in A \text{ and } w \text{ doesn't contain any string in } B \text{ as a substring}\}.$$

Prove that the class of regular languages is closed under the *avoids* operation.

1.71 Let $\Sigma = \{\texttt{0},\texttt{1}\}$.

a. Let $A = \{\texttt{0}^k u \texttt{0}^k|\ k \geq 1 \text{ and } u \in \Sigma^*\}$. Show that A is regular.

b. Let $B = \{\texttt{0}^k \texttt{1} u \texttt{0}^k|\ k \geq 1 \text{ and } u \in \Sigma^*\}$. Show that B is not regular.

1.72 Let M_1 and M_2 be DFAs that have k_1 and k_2 states, respectively, and then let $U = L(M_1) \cup L(M_2)$.

a. Show that if $U \neq \emptyset$, then U contains some string s, where $|s| < \max(k_1, k_2)$.

b. Show that if $U \neq \Sigma^*$, then U excludes some string s, where $|s| < k_1k_2$.

1.73 Let $\Sigma = \{\texttt{0},\texttt{1},\texttt{\#}\}$. Let $C = \{x\texttt{\#}x^{\mathcal{R}}\texttt{\#}x|\ x \in \{\texttt{0},\texttt{1}\}^*\}$. Show that $\overline{C}$ is a CFL.

SELECTED SOLUTIONS

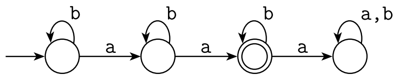
1.1 For M_1: **(a)** q_1; **(b)** $\{q_2\}$; **(c)** q_1, q_2, q_3, q_1, q_1; **(d)** No; **(e)** No
For M_2: **(a)** q_1; **(b)** $\{q_1, q_4\}$; **(c)** q_1, q_1, q_1, q_2, q_4; **(d)** Yes; **(e)** Yes

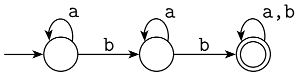
1.2 $M_1 = (\{q_1, q_2, q_3\}, \{\mathtt{a}, \mathtt{b}\}, \delta_1, q_1, \{q_2\})$.
$M_2 = (\{q_1, q_2, q_3, q_4\}, \{\mathtt{a}, \mathtt{b}\}, \delta_2, q_1, \{q_1, q_4\})$.
The transition functions are

δ_1	a	b
q_1	q_2	q_1
q_2	q_3	q_3
q_3	q_2	q_1

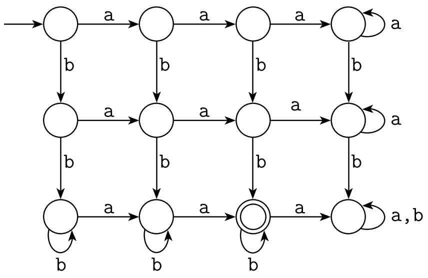
δ_2	a	b
q_1	q_1	q_2
q_2	q_3	q_4
q_3	q_2	q_1
q_4	q_3	q_4.

1.4 **(b)** The following are DFAs for the two languages $\{w \mid w$ has exactly two a's$\}$ and $\{w \mid w$ has at least two b's$\}$.

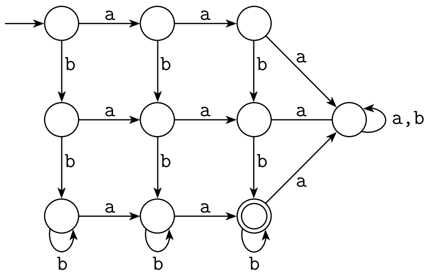




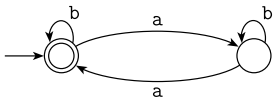
Combining them using the intersection construction gives the following DFA.

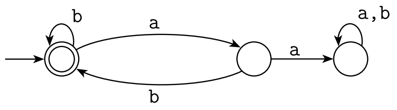


Though the problem doesn't request you to simplify the DFA, certain states can be combined to give the following DFA.

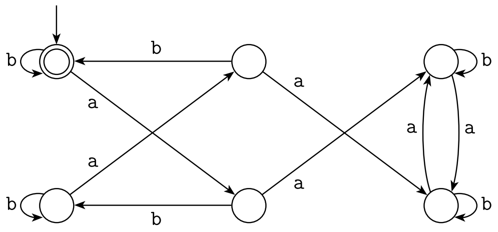


(d) These are DFAs for the two languages $\{w|\ w$ has an even number of a's$\}$ and $\{w|$ each a in w is followed by at least one b$\}$.

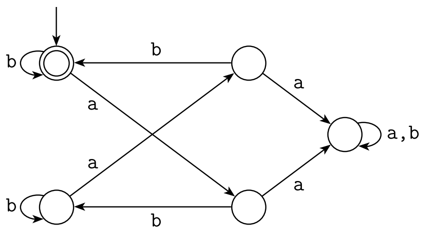




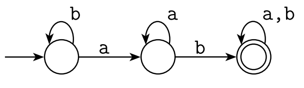
Combining them using the intersection construction gives the following DFA.

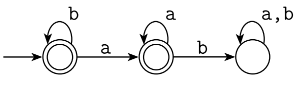


Though the problem doesn't request you to simplify the DFA, certain states can be combined to give the following DFA.

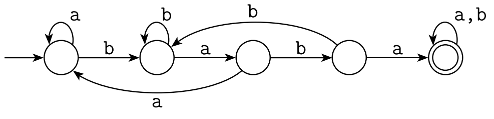


1.5 **(a)** The left-hand DFA recognizes $\{w \mid w \text{ contains } \texttt{ab}\}$. The right-hand DFA recognizes its complement, $\{w \mid w \text{ doesn't contain } \texttt{ab}\}$.

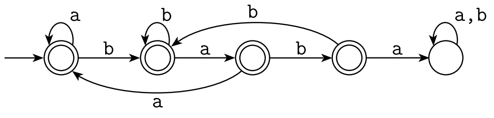




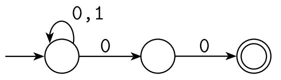
(b) This DFA recognizes $\{w \mid w \text{ contains } \texttt{baba}\}$.



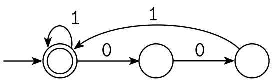
This DFA recognizes $\{w \mid w \text{ does not contain } \texttt{baba}\}$.



1.7 **(a)**



(f)



1.11 Let $N = (Q, \Sigma, \delta, q_0, F)$ be any NFA. Construct an NFA N' with a single accept state that recognizes the same language as N. Informally, N' is exactly like N except it has ε-transitions from the states corresponding to the accept states of N, to a new accept state, q_{accept}. State q_{accept} has no emerging transitions. More formally, $N' = (Q \cup \{q_{\text{accept}}\}, \Sigma, \delta', q_0, \{q_{\text{accept}}\})$, where for each $q \in Q$ and $a \in \Sigma_\varepsilon$

$$\delta'(q,a) = \begin{cases} \delta(q,a) & \text{if } a \neq \varepsilon \text{ or } q \notin F \\ \delta(q,a) \cup \{q_{\text{accept}}\} & \text{if } a = \varepsilon \text{ and } q \in F \end{cases}$$

and $\delta'(q_{\text{accept}}, a) = \emptyset$ for each $a \in \Sigma_\varepsilon$.

1.23 We prove both directions of the "iff."
($\rightarrow$) Assume that $B = B^+$ and show that $BB \subseteq B$.
For every language $BB \subseteq B^+$ holds, so if $B = B^+$, then $BB \subseteq B$.
($\leftarrow$) Assume that $BB \subseteq B$ and show that $B = B^+$.
For every language $B \subseteq B^+$, so we need to show only $B^+ \subseteq B$. If $w \in B^+$, then $w = x_1x_2 \cdots x_k$ where each $x_i \in B$ and $k \geq 1$. Because $x_1, x_2 \in B$ and $BB \subseteq B$, we have $x_1x_2 \in B$. Similarly, because x_1x_2 is in B and x_3 is in B, we have $x_1x_2x_3 \in B$. Continuing in this way, $x_1 \cdots x_k \in B$. Hence $w \in B$, and so we may conclude that $B^+ \subseteq B$.

The latter argument may be written formally as the following proof by induction. Assume that $BB \subseteq B$.
Claim: For each $k \geq 1$, if $x_1, \ldots, x_k \in B$, then $x_1 \cdots x_k \in B$.
Basis: Prove for $k = 1$. This statement is obviously true.
Induction step: For each $k \geq 1$, assume that the claim is true for k and prove it to be true for $k + 1$.
If $x_1, \ldots, x_k, x_{k+1} \in B$, then by the induction assumption, $x_1 \cdots x_k \in B$. Therefore, $x_1 \cdots x_k x_{k+1} \in BB$, but $BB \subseteq B$, so $x_1 \cdots x_{k+1} \in B$. That proves the induction step and the claim. The claim implies that if $BB \subseteq B$, then $B^+ \subseteq B$.

1.29 **(a)** Assume that $A_1 = \{0^n1^n2^n \mid n \geq 0\}$ is regular. Let p be the pumping length given by the pumping lemma. Choose s to be the string $0^p1^p2^p$. Because s is a member of A_1 and s is longer than p, the pumping lemma guarantees that s can be split into three pieces, $s = xyz$, where for any $i \geq 0$ the string xy^iz is in A_1. Consider two possibilities:

1. The string y consists only of 0s, only of 1s, or only of 2s. In these cases, the string $xyyz$ will not have equal numbers of 0s, 1s, and 2s. Hence $xyyz$ is not a member of A_1, a contradiction.
2. The string y consists of more than one kind of symbol. In this case, $xyyz$ will have the 0s, 1s, or 2s out of order. Hence $xyyz$ is not a member of A_1, a contradiction.

Either way we arrive at a contradiction. Therefore, A_1 is not regular.

(c) Assume that $A_3 = \{a^{2^n} \mid n \geq 0\}$ is regular. Let p be the pumping length given by the pumping lemma. Choose s to be the string a^{2^p}. Because s is a member of A_3 and s is longer than p, the pumping lemma guarantees that s can be split into three pieces, $s = xyz$, satisfying the three conditions of the pumping lemma.

The third condition tells us that $|xy| \leq p$. Furthermore, $p < 2^p$ and so $|y| < 2^p$. Therefore, $|xyyz| = |xyz| + |y| < 2^p + 2^p = 2^{p+1}$. The second condition requires $|y| > 0$ so $2^p < |xyyz| < 2^{p+1}$. The length of $xyyz$ cannot be a power of 2. Hence $xyyz$ is not a member of A_3, a contradiction. Therefore, A_3 is not regular.

1.40 **(a)** Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA recognizing A, where A is some regular language. Construct $M' = (Q', \Sigma, \delta', q_0', F')$ recognizing $NOPREFIX(A)$ as follows:

1. $Q' = Q$.
2. For $r \in Q'$ and $a \in \Sigma$, define $\delta'(r, a) = \begin{cases} \{\delta(r, a)\} & \text{if } r \notin F \\ \emptyset & \text{if } r \in F. \end{cases}$
3. $q_0' = q_0$.
4. $F' = F$.

1.44 Let $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ and $M_C = (Q_C, \Sigma, \delta_C, q_C, F_C)$ be DFAs recognizing B and C, respectively. Construct NFA $M = (Q, \Sigma, \delta, q_0, F)$ that recognizes $B \overset{1}{\leftarrow} C$ as follows. To decide whether its input w is in $B \overset{1}{\leftarrow} C$, the machine M checks that $w \in B$, and in parallel nondeterministically guesses a string y that contains the same number of 1s as contained in w and checks that $y \in C$.

1. $Q = Q_B \times Q_C$.
2. For $(q, r) \in Q$ and $a \in \Sigma_\varepsilon$, define

$$\delta((q,r),a) = \begin{cases} \{(\delta_B(q,\mathtt{0}), r)\} & \text{if } a = \mathtt{0} \\ \{(\delta_B(q,\mathtt{1}), \delta_C(r,\mathtt{1}))\} & \text{if } a = \mathtt{1} \\ \{(q, \delta_C(r,\mathtt{0}))\} & \text{if } a = \boldsymbol{\varepsilon}. \end{cases}$$

3. $q_0 = (q_B, q_C)$.
4. $F = F_B \times F_C$.

1.46 **(b)** Let $B = \{\mathtt{0}^m\mathtt{1}^n \mid m \neq n\}$. Observe that $\overline{B} \cap \mathtt{0}^*\mathtt{1}^* = \{\mathtt{0}^k\mathtt{1}^k \mid k \geq 0\}$. If B were regular, then $\overline{B}$ would be regular and so would $\overline{B} \cap \mathtt{0}^*\mathtt{1}^*$. But we already know that $\{\mathtt{0}^k\mathtt{1}^k \mid k \geq 0\}$ isn't regular, so B cannot be regular.

Alternatively, we can prove B to be nonregular by using the pumping lemma directly, though doing so is trickier. Assume that $B = \{\mathtt{0}^m\mathtt{1}^n \mid m \neq n\}$ is regular. Let p be the pumping length given by the pumping lemma. Observe that $p!$ is divisible by all integers from 1 to p, where $p! = p(p-1)(p-2)\cdots 1$. The string $s = \mathtt{0}^p\mathtt{1}^{p+p!} \in B$, and $|s| \geq p$. Thus the pumping lemma implies that s can be divided as xyz with $x = \mathtt{0}^a$, $y = \mathtt{0}^b$, and $z = \mathtt{0}^c\mathtt{1}^{p+p!}$, where $b \geq 1$ and $a+b+c = p$. Let s' be the string $xy^{i+1}z$, where $i = p!/b$. Then $y^i = \mathtt{0}^{p!}$ so $y^{i+1} = \mathtt{0}^{b+p!}$, and so $s' = \mathtt{0}^{a+b+c+p!}\mathtt{1}^{p+p!}$. That gives $s' = \mathtt{0}^{p+p!}\mathtt{1}^{p+p!} \notin B$, a contradiction.

1.50 Assume to the contrary that some FST T outputs $w^{\mathcal{R}}$ on input w. Consider the input strings 00 and 01. On input 00, T must output 00, and on input 01, T must output 10. In both cases, the first input bit is a 0 but the first output bits differ. Operating in this way is impossible for an FST because it produces its first output bit before it reads its second input. Hence no such FST can exist.

1.52 **(a)** We prove this assertion by contradiction. Let M be a k-state DFA that recognizes L. Suppose for a contradiction that L has index greater than k. That means some set X with more than k elements is pairwise distinguishable by L. Because M has k states, the pigeonhole principle implies that X contains two distinct strings x and y, where $\delta(q_0, x) = \delta(q_0, y)$. Here $\delta(q_0, x)$ is the state that M is in after starting in the start state q_0 and reading input string x. Then, for any string $z \in \Sigma^*$, $\delta(q_0, xz) = \delta(q_0, yz)$. Therefore, either both xz and yz are in L or neither are in L. But then x and y aren't distinguishable by L, contradicting our assumption that X is pairwise distinguishable by L.

(b) Let $X = \{s_1, \ldots, s_k\}$ be pairwise distinguishable by L. We construct DFA $M = (Q, \Sigma, \delta, q_0, F)$ with k states recognizing L. Let $Q = \{q_1, \ldots, q_k\}$, and define $\delta(q_i, a)$ to be q_j, where $s_j \equiv_L s_i a$ (the relation $\equiv_L$ is defined in Problem 1.51). Note that $s_j \equiv_L s_i a$ for some $s_j \in X$; otherwise, $X \cup s_i a$ would have $k+1$ elements and would be pairwise distinguishable by L, which would contradict the assumption that L has index k. Let $F = \{q_i \mid s_i \in L\}$. Let the start state q_0 be the q_i such that $s_i \equiv_L \boldsymbol{\varepsilon}$. M is constructed so that for any state q_i, $\{s \mid \delta(q_0, s) = q_i\} = \{s \mid s \equiv_L s_i\}$. Hence M recognizes L.

(c) Suppose that L is regular and let k be the number of states in a DFA recognizing L. Then from part (a), L has index at most k. Conversely, if L has index k, then by part (b) it is recognized by a DFA with k states and thus is regular. To show that the index of L is the size of the smallest DFA accepting it, suppose that L's index is *exactly* k. Then, by part (b), there is a k-state DFA accepting L. That is the smallest such DFA because if it were any smaller, then we could show by part (a) that the index of L is less than k.

1.55 **(a)** The minimum pumping length is 4. The string `000` is in the language but cannot be pumped, so 3 is not a pumping length for this language. If s has length 4 or more, it contains `1`s. By dividing s into xyz, where x is `000` and y is the first `1` and z is everything afterward, we satisfy the pumping lemma's three conditions.

(b) The minimum pumping length is 1. The pumping length cannot be 0 because the string ε is in the language and it cannot be pumped. Every nonempty string in the language can be divided into xyz, where x, y, and z are ε, the first character, and the remainder, respectively. This division satisfies the three conditions.

(d) The minimum pumping length is 3. The pumping length cannot be 2 because the string `11` is in the language and it cannot be pumped. Let s be a string in the language of length at least 3. If s is generated by $\texttt{0}^*\texttt{1}^+\texttt{0}^+\texttt{1}^*$ and s begins either `0` or `11`, write $s = xyz$ where $x = \varepsilon$, y is the first symbol, and z is the remainder of s. If s is generated by $\texttt{0}^*\texttt{1}^+\texttt{0}^+\texttt{1}^*$ and s begins `10`, write $s = xyz$ where $x = \texttt{10}$, y is the next symbol, and z is the remainder of s. Breaking s up in this way shows that it can be pumped. If s is generated by $\texttt{10}^*\texttt{1}$, we can write it as xyz where $x = \texttt{1}$, $y = \texttt{0}$, and z is the remainder of s. This division gives a way to pump s.