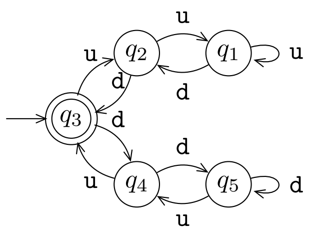
1. The formal description of a DFA M is $(\{q_1, q_2, q_3, q_4, q_5\}, \{u, d\}, \delta, q_3, \{q_3\})$, where δ is given by the following table. Give the state diagram of this machine.

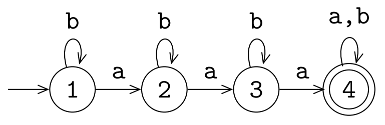
	u	d
q_1	q_1	q_2
q_2	q_1	q_3
q_3	q_2	q_4
q_4	q_3	q_5
q_5	q_4	q_5

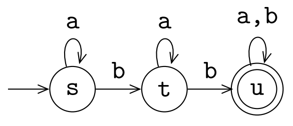
Answer:

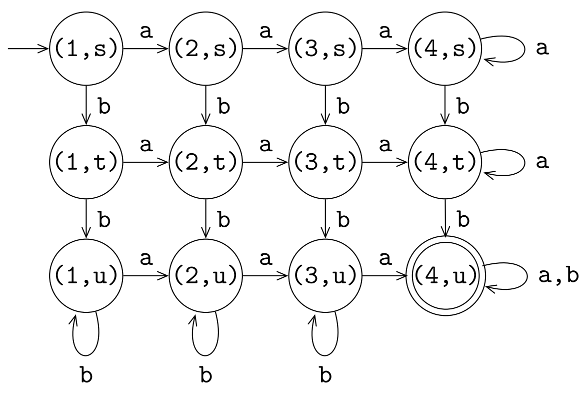


2. Each of the following languages is the intersection of two simpler languages. In each part, construct DFAs for the simpler languages, then combine them using the construction discussed in footnote 3 (page 46) to give the state diagram of a DFA for the language given. In all parts, $\Sigma = \{a, b\}$.

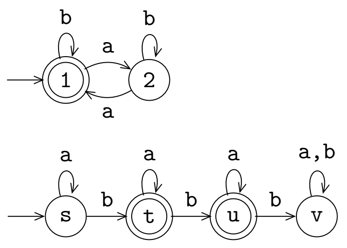
a. $\{w \mid w$ has at least three a's and at least two b's $\}$
Answer:

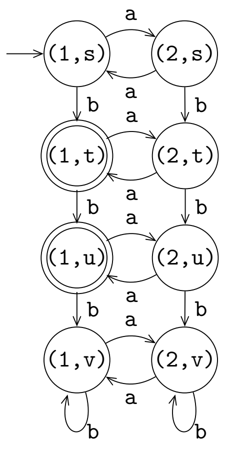




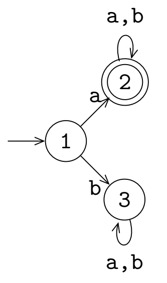


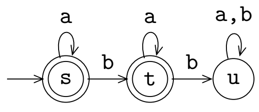
c. $\{w \mid w$ has an even number of a's and one or two b's $\}$
Answer:

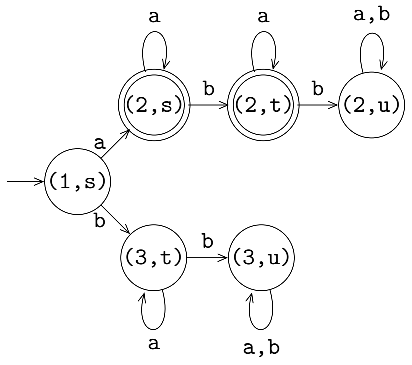




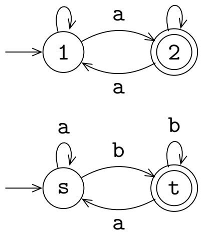
e. $\{w \mid w$ starts with an a and has at most one b $\}$
Answer:

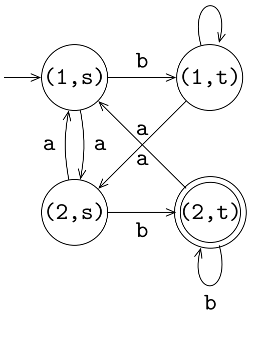






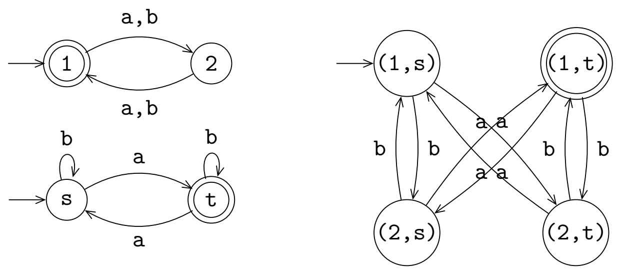
f. $\{w \mid w$ has an odd number of a's and ends with a b $\}$
Answer:



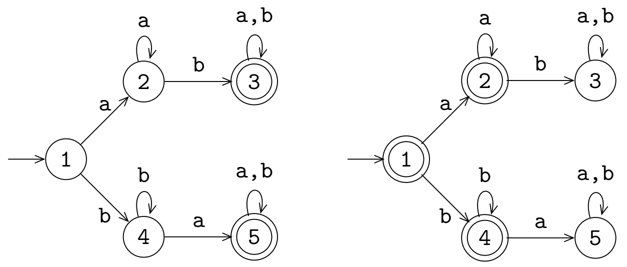


g. $\{w \mid w$ has even length and an odd number of a's $\}$

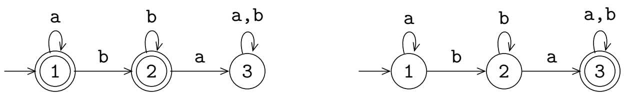
Answer:



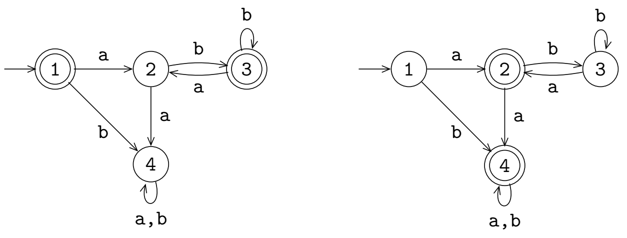
3. Each of the following languages is the complement of a simpler language. In each part, construct a DFA for the simpler language, then use it to give the state diagram of a DFA for the language given. In all parts, $\Sigma = \{a, b\}$.
c. $\{w \mid w$ contains neither the substrings ab nor ba $\}$
Answer:



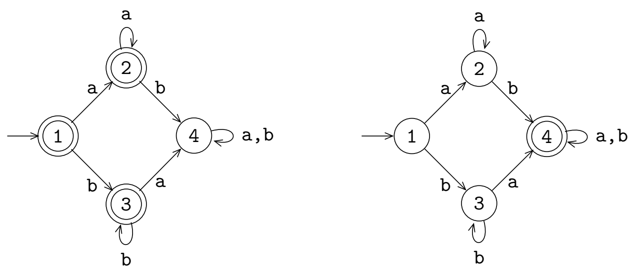
d. $\{w \mid w$ is any string not in a*b* $\}$
Answer:



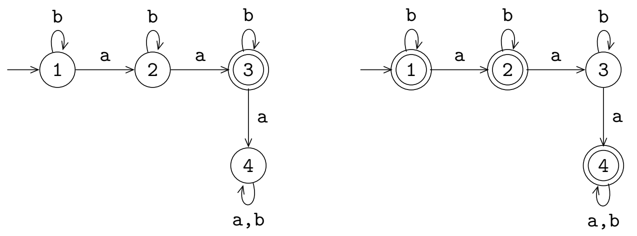
e. $\{w \mid w$ is any string not in (ab$^+$)* $\}$
Answer:



f. $\{w \mid w$ is any string not in a*∪b* $\}$
Answer:

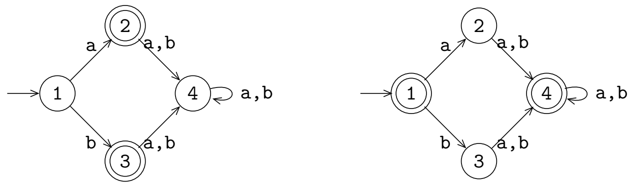


g. $\{w \mid w$ is any string that doesn't contain exactly two a's $\}$
Answer:

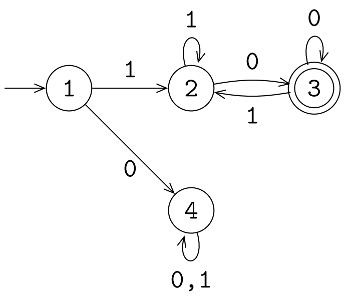


h. $\{w \mid w$ is any string except a and b $\}$

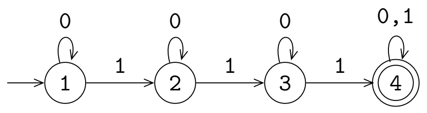
Answer:



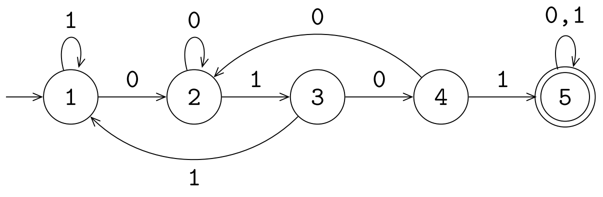
4. Give state diagrams of DFAs recognizing the following languages. In all parts, the alphabet is $\{0, 1\}$.
a. $\{w \mid w$ begins with a 1 and ends with a 0$\}$
Answer:



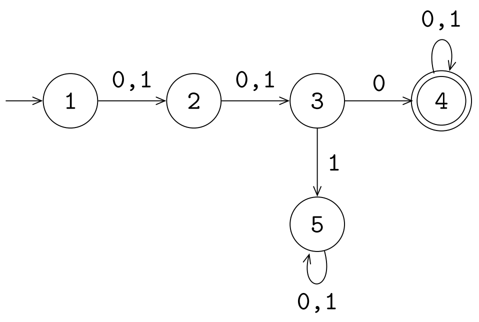
b. $\{w \mid w$ contains at least three 1s$\}$
Answer:



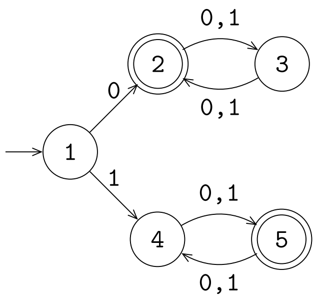
c. $\{w \mid w$ contains the substring 0101 (i.e., $w = x0101y$ for some x and y)$\}$
Answer:



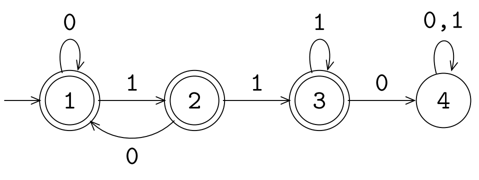
d. $\{w \mid w$ has length at least 3 and its third symbol is a 0$\}$
Answer:



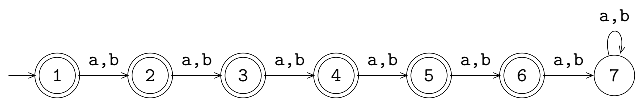
e. $\{w \mid w$ starts with 0 and has odd length, or starts with 1 and has even length$\}$
Answer:



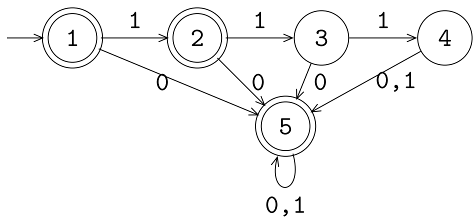
f. $\{w \mid w$ doesn't contain the substring 110$\}$
Answer:



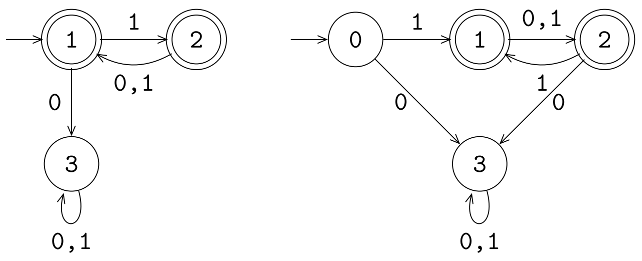
g. $\{w \mid$ the length of w is at most 5$\}$
Answer:



h. $\{w \mid w$ is any string except 11 and 111$\}$
Answer:

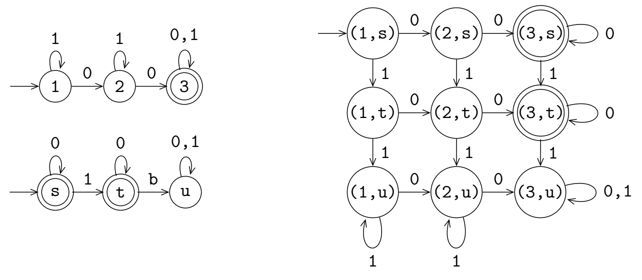


i. $\{w \mid$ every odd position of w is a 1$\}$
Answer:

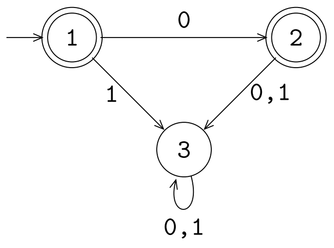


not including ε

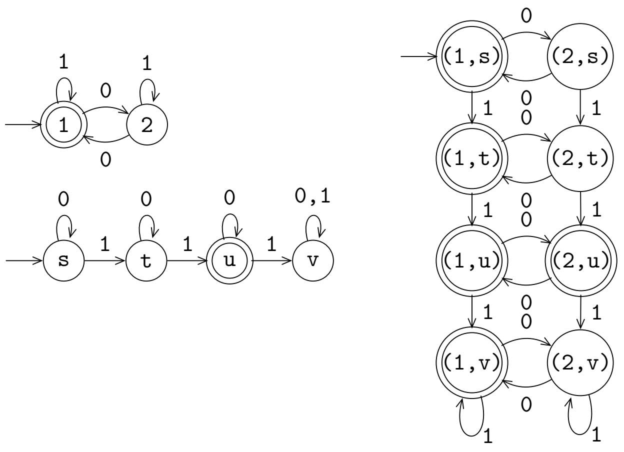
j. $\{w \mid w$ contains at least two 0s and at most one 1$\}$
Answer:



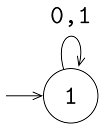
k. $\{\varepsilon,0\}$
Answer:



l. $\{w \mid w$ contains an even number of 0s, or contains exactly two 1s$\}$
Answer:

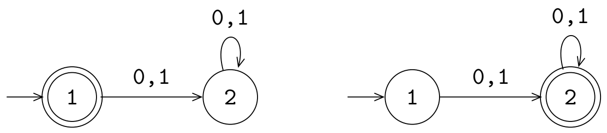


m. The empty set
Answer:

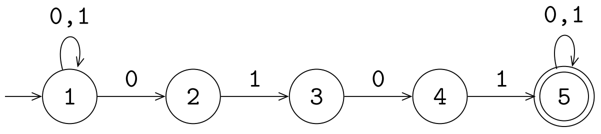


n. All strings except the empty string

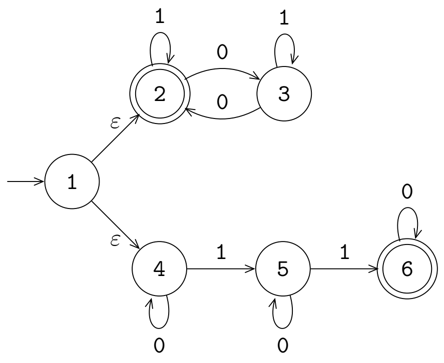
Answer:



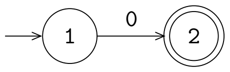
5. Give state diagrams of NFAs with the specified number of states recognizing each of the following languages. In all parts, the alphabet is $\{0, 1\}$.
b. The language $\{w \mid w$ contains the substring 0101 (i.e., $w = x0101y$ for some x and y)$\}$ with five states
Answer:



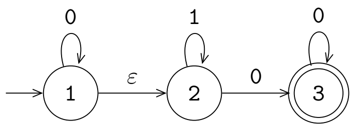
c. The language $\{w \mid w$ contains an even number of 0s, or contains exactly two 1s$\}$ with six states
Answer:



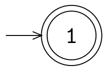
d. The language $\{0\}$ with two states
Answer:



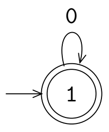
e. The language $0^*1^*0^+$ with three states
Answer:



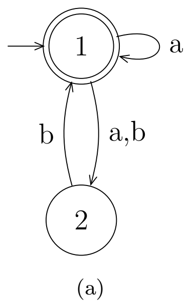
g. The language $\{\varepsilon\}$ with one state
Answer:



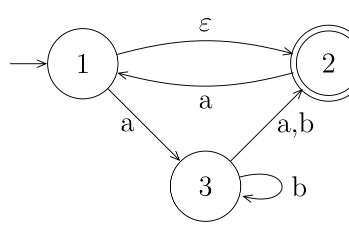
h. The language 0^* with one state **Answer:**



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



(a)

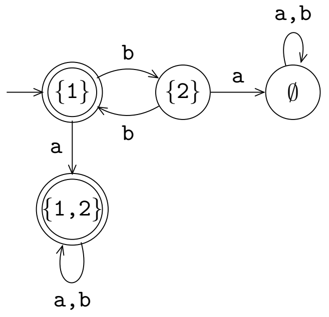


(b)

Answer:

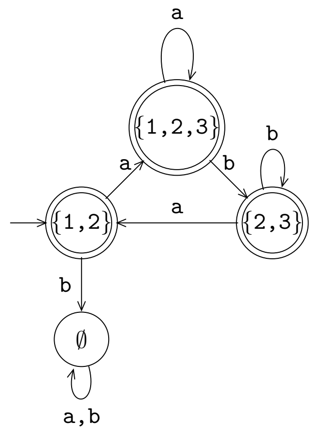
(a) δ:

	a	b
{1}	{1,2}	{2}
{2}	∅	{1}
{1,2}	{1,2}	{1,2}



(b) δ:

	a	b
{1,2}	{1,2,3}	∅
{1,2,3}	{1,2,3}	{2,3}
{2,3}	{1,2}	{2,3}



7. For any string $w = a_1a_2\cdots a_n$, the reverse of w, written $w^{\mathcal{R}}$, is the string w in reverse order, $a_n\cdots a_2a_1$. For any language A, let $A^{\mathcal{R}} = \{w^{\mathcal{R}} \mid w \in A\}$. Prove that if A is regular, so is $A^{\mathcal{R}}$.

Answer 1:

Assume that A is regular. Then there exists an NFA $M = (Q, \Sigma, \delta, q_s, F)$, such that $L(M) = A$. Produce a new NFA M' using the following process:

(a) Let $M' = M = (Q', \Sigma, \delta', q'_s, F')$
(b) Set $Q' = Q \cup \{q_{new}\}$
(c) Set $F' = \{q_s\}$
(d) Set $q'_s = q_{new}$
(e) Remove each transition rule $\delta'(q, \sigma) = p$ and replace it with $\delta'(p, \sigma) = q$ for $\sigma \in \Sigma \cup \varepsilon$ (e.g. remove all the arrows).
(f) $\forall q \in F$ set $\delta'(q_{new}, \varepsilon) = q$

We now argue that $L(M') = A^{\mathcal{R}}$.

Proof. We use the stategy of proving both $L(M') \subset A^{\mathcal{R}}$ and $A^{\mathcal{R}} \subset L(M')$.

- $L(M') \subset A^{\mathcal{R}}$
  Assume $w \in L(M')$ then $\exists$ a sequence of states $r = q'_s, r_0, \ldots, r_n$ in Q' such that the first state is $q'_s, \delta'(r_i, w_{i+1}) = r_{i+1}$, for $i = 0, \ldots, n-1$ (also $\delta'(q'_s, \varepsilon) = r_0$), and $r_n \in F'$. But $F' = \{q_s\}$ so $r_n = q_s$, and $\delta'(r_i, w_{i+1}) = r_{i+1}$ implies $\delta(r_{i+1}, w_{i+1}) = r_i$ because δ' was constructed by reversing transitions of δ. Finally q'_s has an epsilon transition to each $q \in F$, so $r_0 \in F$. Therefore by reversing the sequense r (without the leading q'_s) we have a computation for $w^{\mathcal{R}}$ on machine M, so M accepts $w^{\mathcal{R}}$, so $w^{\mathcal{R}\mathcal{R}} = w \in A^{\mathcal{R}}$.
- $A^{\mathcal{R}} \subset L(M')$
  Assume $w \in A^{\mathcal{R}}$. Then $w^{\mathcal{R}} \in A$ which means $w^{\mathcal{R}} \in L(M)$, so there $\exists$ a sequence of states $r = r_0, \ldots, r_n$ in Q such that $r_0 = q_s, \delta(r_i, w_{n-i+1}) = r_{r+1}$, for $i = 0, \ldots, n-1$, and $r_n \in F$. Following the construction above we can find a new sequence $r'_0, \ldots, r'_n$ where $r'_0 \in F, r'_n \in F'$, and $\delta'(r'_i, w_{i+1}) = r'_{i+1}$. By taking $r' = r^{\mathcal{R}}$ e.g. the reversal of r. Finally we add q'_s to the beginning of this sequence and note that $\delta'(q'_S, \varepsilon) = r'_0$ since $r'_0 \in F$. Now the full sequence $q'_s, r'_0, \ldots, r'_n$ is a computation for the string w on machine M', so $w \in L(M')$.

Answer 2:

Recursively (or inductively) define a reversing operation on regular expressions, and apply that operation on the regular expression for A. In particular, given a regular expression R, `reverse`(R) is:

- a for some $a \in \Sigma$,
- ε if $R = \varepsilon$,
- $\emptyset$ if $R = \emptyset$,
- $(\texttt{reverse}(R_1) \cap \texttt{reverse}(R_2))$, if $R = R_1 \cap R_2$,
- $(\texttt{reverse}(R_2) \circ \texttt{reverse}(R_1))$, if $R = R_1 \circ R_2$, or
- $(\texttt{reverse}(R_1)^*)$, if $R = (R_1^*)$.

8. For languages A and B, let the ***perfect shuffle*** of A and B be the language $\{w \mid w = a_1b_1 \cdots a_kb_k$, where $a_1 \cdots a_k \in A$ and $b_1 \cdots b_k \in B$, each $a_i, b_i \in \Sigma\}$. Prove that the class of regular languages is closed under perfect shuffle.

   Answer

   Let $N_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1)$ and $N_2 = (Q_2, \Sigma_2, \delta_2, q_2, F_2)$ be two DFAs that recognize A and B, respectively. We design an NFA $N = (Q, \Sigma, \delta, q_0, F)$ that recognizes the shuffle of A and B instead of directly designing a DFA.

   At any time, N needs to keep track of the current states of A and B accordingly. After the whole string is processed, if both DFAs are in the accept states, the input string is accepted; otherwise, the input string is rejected. In addition, N should also accept the empty string.

   The δ function for this DFA will use δ_1(from the DFA that recognizes A) when the third character of the state is a '1', and will use δ_2 when the character is a '2'.

   Formally, the NFA N can be difined as follows:

   (a) $Q = Q_1 \times Q_2 \times \{1, 2\}$

   (b) $q = (q_{A_0}, q_{B_0}, 1)$.

   (c) $F = F_1 \times F_2 \times \{2\}$

   (d) $\Sigma = \Sigma_1 \cup \Sigma_2$

   (e) δ is as follows:

      i. $\delta((q_{A_i}, q_{B_j}, 1), \sigma) = (\delta_1(q_{A_i}, a), q_{B_j}, 2)\delta((q_{A_i}, q_{B_j}, 1), \sigma) = (\textit{empty set})\sigma \notin \Sigma_1$

      ii. $\delta((q_{A_i}, q_{B_j}, 2), \sigma) = (q_{A_i}, \delta_2(q_{B_i}, a), 1)\delta((q_{A_i}, q_{B_j}, 2), \sigma) = (\textit{empty set})\sigma \notin \Sigma_2$

      iii. $q_{A_i} \in Q_1$

      iv. $q_{B_j} \in Q_2$

      v. $\sigma \in \Sigma$

   - $L(N) \subseteq \text{PerfectShuffle}(A, B)$

     Let $w \in L(N)$. Then w must go through states$q_{0_1}, q_{i_2}, \ldots, q_{i_k}$ from N_1 and $r_{0_1}, \ldots, r_{i_k}$ from N_2. Then the sequence of states for N to recognize PerfectShuffle(x, y) is the start state:

     $\{q_{0_1}, r_{0_1}, 1\}$

     and by the definition of δ and δ_1 the next state is:

     $\{q_{i_2}, r_{0_1}, 2\}$

     and by definition of δ and δ_2 the next state is:

     $\{q_{i_2}, r_{i_2}, 1\}$

     ...keep alternation the use of δ_1 and δ_2 until we get to:

     $\{q_{i_k}, r_{i(k-1)}, 1\}$(the next to last state)

     and by the definition of δ and δ_1 the next state is:

     $\{q_{i_k}, r_{i_k}, 2\}$(accepting state)

     So $w = a_1b_1a_2b_2 \ldots a_kb_k$ where $a_1 \ldots a_k \in A$ and $b_1 \ldots b_k \in B$ implies $w \in \text{PerfectShuffle}(A, B)$

     So $L(N) \subseteq \text{PerfectShuffle}(A, B)$

   - $\text{PerfectShuffle}(A, B) \subseteq L(N)$

     Assume A and B not empty (otherwise this case is trivial) by the definition of $\text{PerfectShuffle}(A, B)$

     and any $w \in \text{PerfectShuffle}(A, B)$ has an even number of character from Σ

     For any strings $s_i \in A$ and $r_i \in B$ used to form w, where $|s_i| = |r_i| = k$ and $w = s_{i_1}r_{i_1}s_{i_2} \ldots r_{i_k}$. All w's will be in $\text{PerfectShuffle}(A, B)$.

     And there w's will be recognized by N.

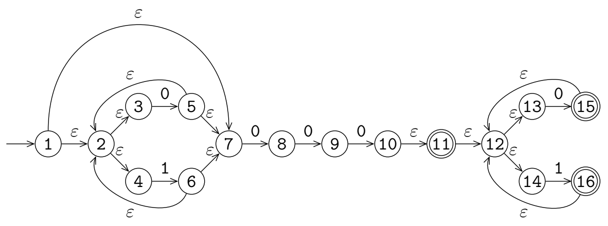
     So $\text{PerfectShuffle}(A, B) \subseteq L(N)$

   Therefore, $\text{PerfectShuffle}(A, B)$ is a regular language. Any regular language A and B are closed under $\text{PerfectShuffle}(A, B)$.

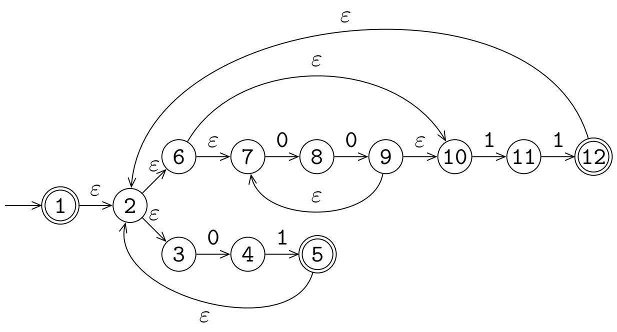
9. Use the procedure described in Lemma1.55 to convert the following regular expressions to nondeterministic finite automata.

   a. $(0 \cup 1)^*000(0 \cup 1)^*$

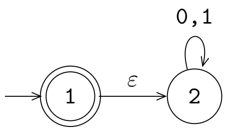
   Answer:



b. $(((00)^*(11)) \cup 01)^*$
Answer:



c. $\emptyset^*$
Answer:



10. 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^*
Answer:

ab,abb;ba,aba

b. $a(ba)^*b$
Answer:

ab,abab;a,b

c. $a^* \cup b^*$
Answer:

a,b;ab,ba

d. $(aaa)^*$
Answer:

aaa,aaaaaa;a,aa

e. $\Sigma^*a\Sigma^*b\Sigma^*a\Sigma^*$
Answer:

aba,aaba;a,b

f. aba∪bab
Answer:

aba,bab;a,b

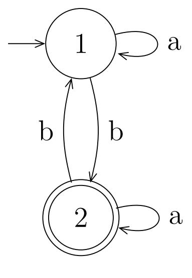
g. $(\varepsilon \cup a)b$
Answer:

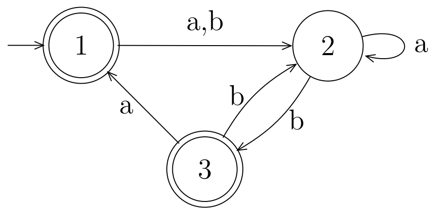
ab,b;a,aa

h. $(a \cup ba \cup bb)\Sigma^*$
Answer:

a,ba;ε,b

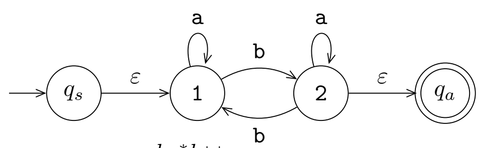
11. Use the procedure described in Lemma 1.60 to convert the following finite automata to regular expressions.

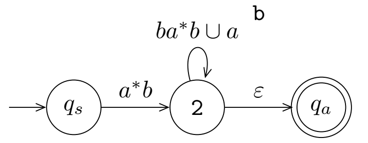

(a)

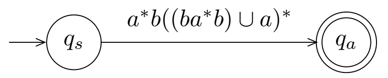

(b)

Answer:

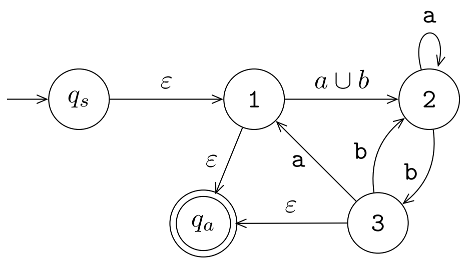
(a)

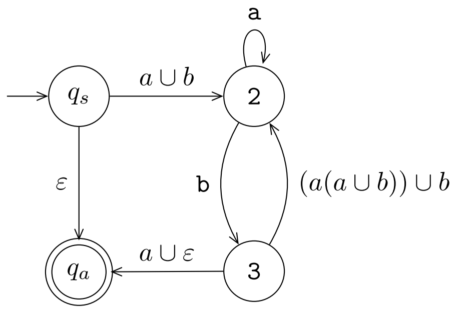


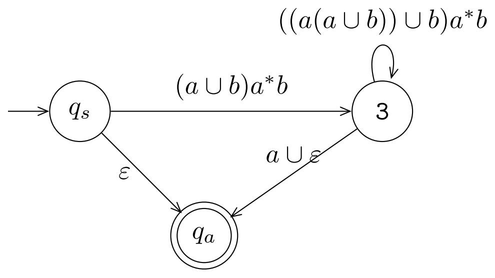


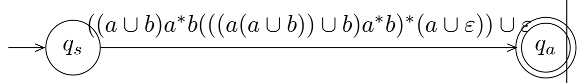


(b)

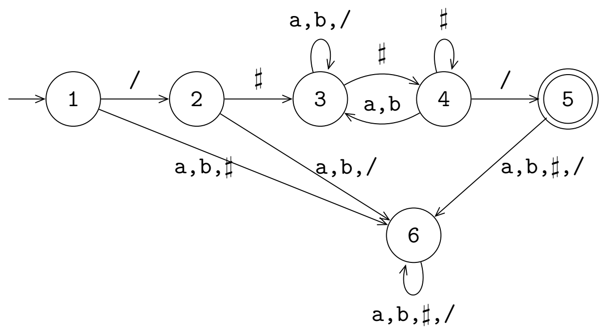







$((a \cup b)a^*b(((a(a \cup b)) \cup b)a^*b)^*(a \cup \varepsilon)) \cup \varepsilon$
q_s
q_a

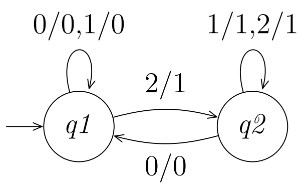
12. In certain programming languages, comments appear between delimiters such as /# and #/. Let C be the language of all valid delimited comment strings. A member of C must begin with /# and end with #/ but have no intervening #/. For simplicity, assume that the alphabet for C is Σ={a,b,/,#}.
a. Give a DFA that recognizes C.
Answer:



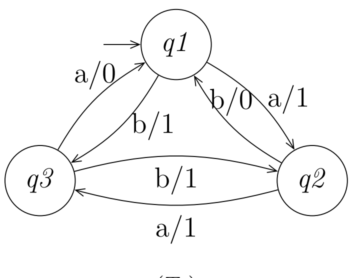
b. Give a regular expression that generates C.
Answer:

$$/\sharp(a \cup b \cup /)^*\sharp(((a \cup b)(a \cup b \cup /)^*\sharp) \cup \sharp)^*/$$

13. A ***finite state transducer*** (FST) is a type of deterministic finite automat on whose out put 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 $w_1 \ldots w_n$ one by one and, starting at the start state, follows the transitions by matching the input labels with the sequence of

symbols $w_1 \dots w_n = w$. Every time it goes along a transition, it outputs the corresponding out put 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

Answer:

	states sequence	output
a	q_1, q_1, q_1, q_1	000
b	q_1, q_2, q_2, q_2	111
c	q_1, q_1, q_2, q_2	011
d	q_1, q_1, q_2, q_1, q_2	0101
e	q_1, q_3	1
f	q_1, q_3, q_2, q_3, q_2	1111
g	$q_1, q_3, q_2, q_1, q_3, q_2, q_1$	110110
h	q_1	ε

14. Read the informal definition of the finite state transducer given in the previous exercise. 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$\delta : Q \times \Sigma \longrightarrow Q \times \Gamma$.)

Answer:

A finite state transducer (FST) is a 5-tuple $(Q, \Sigma, \delta, q_0, \Gamma)$, where
1. Q is a finite set called the *states*,
2. Σ is a finite set called the *input alphabet*,
3. $\delta : Q \times \Sigma \to Q \times \Gamma$ is the *transition function*,
4. $q_0 \in Q$ is the *start state*, and
5. Γ is a finite set called the *output alphabet*.

A computation of an FST $F = (Q, \Sigma, \delta, q_0, \Gamma)$ on input $w = w_1 w_2 ... w_n \in \Sigma^*$ in Q such that

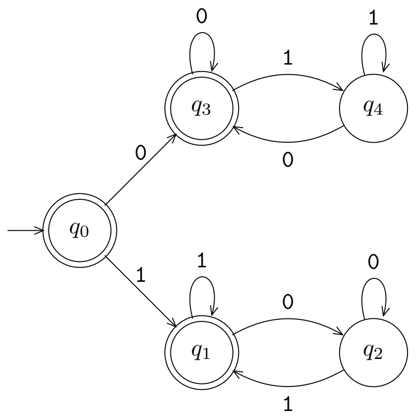
1. $r_0 = q_0$, and
2. $\delta(r_i, w_{i+1}) = (r_{i+1}, x_{i+1})$ for $0 \le i \le n$.

15. Let $\Sigma = \{0, 1\}$ and let

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

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

Answer:



To formally prove the correctness of this construction, we must show that $L(M) = A$, where M is the constructed DFA and A is the language described in the problem.

Proof. ($L(M) = A$) We prove this conjecture by showing $L(M) \subseteq A$ and also $A \subseteq L(M)$, which implies that $L(M) = A$.

We first need to argue that the number of ‘01’ and ‘10’ substrings of a binary string are equal iff the first and last symbol of the string are equal.

Lemma. *Let w be an arbitrary length n binary string. Define mappings from $\{0,1\}^*$ to $\mathbb{N}$: $n_{10}(w)$ = (# of ‘10’ substrings in w) and $n_{01}(w)$ = (# of ‘01’ substrings in w). We conjecture that $|n_{01}(w) - n_{10}(w)| \le 1 \ \forall w \in \{0,1\}^*$, and that $n_{01}(w) = n_{10}(w)$ if and only if $w_0 = w_n$ (first and last symbol same).*

Proof. Create w' from w by replacing all strings of consecutive 0s in w by a single 0 and all strings of consecutive 1s in w by a single 1. Now consider all overlapping pairs $(w'_0, w'_1), (w'_1, w'_2), \ldots, (w'_{n-2}, w'_{n-1}), (w'_{n-1}, w'_n)$. Each pair is a '01' and '10' substring since there are no consecutive symbols, and adjacent pairs connot be the same substring because then the inner symbol would be 0 and 1 at the same time (e.g. (w_{i-1}, w_i) and (w_i, w_{i+1}) cnanot be '01' and '01' or '10' and '10'.) Therefore it must be that $|n_{01}(w') - n_{10}(w')| \leq 1$, and since w is simply w' with added strings of consecutive symbol, which would not affect the number of '01' or '10' substrings, we conclude that $|n_{01}(w) - n_{10}(w)| \leq 1$. Now note that in the special case that $w_0 = w_n$, then $w'_0 = w'_n$, so one of (w'_0, w'_1) and (w'_{n-1}, w'_n) is a '01' substring and one is a '10' substring. Since all adjacent overlapping pairs are different, $n_{10}(w') = n_{01}(w')$ and therefore $n_{10}(w) = n_{01}(w)$. Conversely, if $n_{10}(w) = n_{01}(w)$, then $n_{10}(w') = n_{01}(w')$, which means there must be an even number of overlapping pairs. Since adjacent pairs are different substrings, the first and last pairs must be different, meaning one is '01' and one is '10'. Either way, $w'_0 = w'_n$ and therefore $w_0 = w_n$.

Now we use the lemma to prove the main result.

- $(L(M) \subseteq A)$

  Suppose $w \in L(M)$ where w is a binary string. Then w is accepted by machine M, which means $\exists$ a sequence of states $r = r_0, \ldots, r_n$ in $Q(M)$ (set of states of machine M). Such that $r_0 = q_0, \delta(r_i, w_{i+1}) = r_{i+1}$, for $i = 0, \ldots, n-1$, and $r_n \in F$. Assume WLOG $w_0 = 1$, then $r_1 = q_1$. Also, $r_2, \ldots, r_n \in \{q_1, q_2\}$ because those are the only states reachable form q_1. But $r_n \in F$, so $r_n \in F \cap \{q_1, q_2\} = \{q_1\}$. This means $w_n =$'1', since q_1 is only reachable by transitions on '1'. By the lemma, since $w_0 = 0$, where you can argue that $w_n = 0$ and then use the lemma.

- $(A \subseteq L(M))$

  Suppose $w \in A$. Assume WLOG $w_0 = 1$. Create a sequence of states $\{r_i\}_{i=1}^n$ where $r_i = q_1$ if $w_i = 1$ and $r_i = q_2$ if $w_i = 0$. We now take $r_0 = q_0$ and see that r_0 is the start state of M, $\delta(r_i, w_{r+1}) = r_{i+1}, \forall i = 1, \ldots, n-1$ which was forced on the created sequence. Finally since $w \in A$, which implies $w_0 = w_n$ by the lemma, we have that $w_n = 1$ and therefore $r_n = q_1 \in F$. This means that M accepts $w \in L(M)$.

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

    b. $A_2 = \{www \mid w \in \{a, b\}^*\}$

    Answer:

    To prove that A_2 is not a regular language, we will use a proof by contradiction. Assume that A_2 is regular. Then by the Pumping Lemma for Regular Languages, there exists a pumping length, p for A_2 such that for any string $s \in A_2$ where $|s| \geq p$, $s = xyz$ subject to the following conditions:
    (a) $|y| > 0$
    (b) $|xy| \leq p$
    (c) $\forall i > 0, xy^iz \in A_2$.
    Choose $s = a^pba^pba^pb$. Clearly, $|s| \geq p$ and $s \in A_2$. By condition (b) above, it follows that x and y are composed only of a. By condition (a), it follows that $y = a^k$ for some $k > 0$. By condition (c), we can take $i = 2$ and the resulting string will still be in A_2. Thus, xy^2z should be in A_2. $xy^2z = a^{p+k}ba^pba^b$. But, this is clearly not in L. This is a contradiction with the pumping lemma. Therefore our assumption that A_2 is regular is incorrect and A_2 is not a regular language.

17. 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. $\{0^n1^m0^n \mid m, n \geq 0\}$
    Answer:

    To prove that L is not a regular language, we will use a proof by contradiction. Assume that L is regular. Then by the Pumping Lemma for Regular Languages, there exists a pumping length,p for L such that for any string $s \in L$ where $|s| \geq p$, $s = xyz$ subject to the following conditions:
    (a) $|y| > 0$
    (b) $|xy| \leq p$

(c) $\forall i > 0, xy^iz \in L$.
Choose $s = 0^p10^p$. Clearly, $|s| \geq p$ and $s \in L$. By condition (b) above, it follows that x and y are composed only of zeros. By condition (a), it follows that $y = 0^k$ for some $k > 0$. Per (c), we can take $i = 0$ and the resulting string will still be in L. Thus, xy^0z should be in L. $xy^0z = xz = 0^{p-k}10^p$. But, this is clearly not in L. This is a contradiction with the pumping lemma. Therefore our assumption that L is regular is incorrect and L is not a regular language.

c. $\{w \mid w \in \{0,1\}^*$ is not a palindrome$\}$
Answer:

To show L is not regular, we will use closure properties of regular languages and the pumping lemma. For a contradiction, assume that L is regular. Then $\overline{L} = \{w|w \in \{0,1\}^* is\ a\ palindromeg\}$, L's complement, is regular. We will now use the pumping lemma to show that L is not regular. L has some pumping length p. Let $s = 0^p10^p$. Then $s \in L$ and $|s| > p$. From the pumping lemma, then there must exist some x, y, z such that $s = xyz$ with (1)$|y| > 0$, (2)$|xy| \leq p$, and (3) for each $i \geq 0$,$xy^iz \in \overline{L}$. From (1) and (2) it follows that $y = 0^k$ for some $0 < k \leq p$. However, then $xy^2z = 0^{p+k}10^p \notin L$, since there are more zeros at the start of the string (since $k \neq 0$) than at the end and so it is not a palindrome. This is a contradiction of (3). Thus, we conclude that L is not regular since assuming it was led to the contradiction. (This also shows that the language of palindromes over$0, 1$ is not a regular language).

d. $\{wtw \mid w, t \in \{0,1\}^+\}$
Answer:

To show L is not regular, we will use the pumping lemma. For a contradiction, assume that L is regular. Then L has some pumping length p. Let $s = 0^p110^p1$. Then $s \in L$, with for example $w = 0^p1$ and $t = 1$, and $|s| > p$. From the pumping lemma, then there must exist some x, y, z such that $s = xyz$ with (1)$|y| > 0$, (2) $|xy| \leq p$, and (3) for each $i \neq 0$,$xy^iz \in A$. From (1) and (2) it follows that $y = 0^k$ for some $0 < k \leq p$. However, then $xy^2z = 0^{p+k}110^p1$.For a contradiction, assume that $0^{p+k}110^p1$ is in L, that is can be written as wtw. Then w must end with a single 1 from the last part of the string. Therefore, from the first part of the string we must have that $w = 0^{p+k}1$. However, there are only p 0s at the end of the string before the 1, so w cannot be equal to $0^{p+k}1$. This is a contradiction. Thus, $xy^2z = 0^{p+k}110^p1 \notin L$. This then is a contradiction of (3). Thus, we conclude that L is not regular.

18. Consider the language $F = \{a^ib^jc^k \mid i, j, k \geq 0$ and if $i = 1$ 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.

Answer:

(a) Suppose on the contrary that F is regular. Let $L = \{x \mid x\ begins\ with\ one\ \texttt{a}\}$. Obviously, L is regular. The intersection of two regular languages is regular, so the language $L' = F \cap L$ is regular.

Let p be the pumping leangth of L'. Note that $L' = \{\texttt{a}\texttt{b}^n\texttt{c}^n \mid n \geq 0\}$, so that $\texttt{a}\texttt{b}^p\texttt{c}^p$ is in L'. By pumping lemma, $\texttt{a}\texttt{b}^p\texttt{c}^p$ can be written as xyz such that $|y| > 0$, $|xy| \leq p$, and for each $i \geq 0$, $xy^iz \in L'$. However, (i) if y includes `a`, the string $xyyz$ has at least two `a`'s; (ii) else, if y includes both `b` and `c`, the string $xyyz$ has at least two substring `bc`; (iii) else, y includes only `b` or `c`, and so the number of `b` and the number of `c` in $xyyz$ will not be equal. In all the above three cases, $xyyz$ cannot be in L', so that a contradiction occurs. Thus, F is not regular.

(b) Let p' be the pumping length. Let $p = \texttt{max}\{p', 3\}$ and s be a string in F with $|s| > p$. We divide the discussion into two cases such that Case 1 corresponds to those string s that has exactly two '`a`'s (i.e., when $i = 2$), and Case 2 corresponds to the other kinds of s (i.e., when $i \neq 2$). For Case 1, we can divide s into xyz, such that $x = \varepsilon, y = \texttt{aa}$, and z is the remaining string. For Case 2, we can divide s into

xyz, such that $x = \varepsilon$, y is the first character, and z is the remainder string. We can easily verify that in both cases, $|y| > 0$, $|xy| \leq p$, and for all $i \geq 0$, $xy^iz \in F$.

(c) All regular language satisfies the pumping lemma, but the converse is not true. That is, if a language satisties the pumping lemma, the language may not be regular.