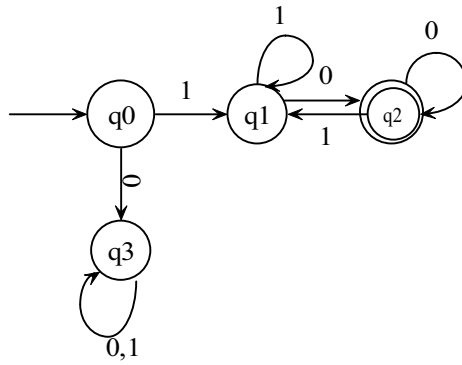
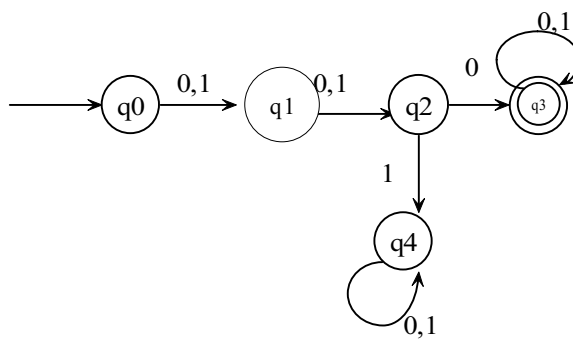


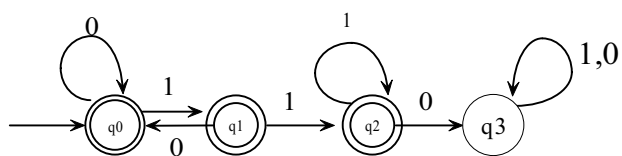
1) 1.4.a



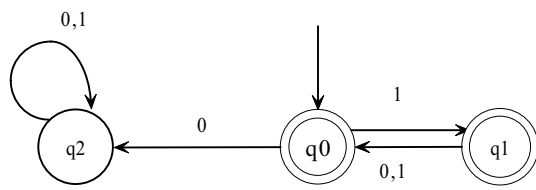
d.



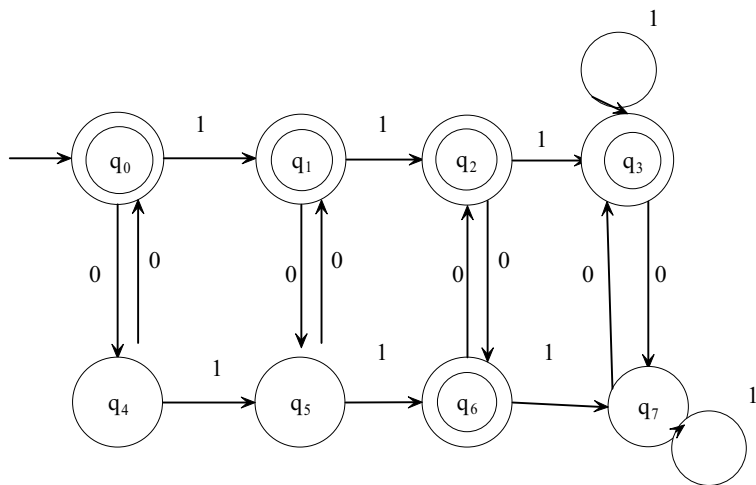
f



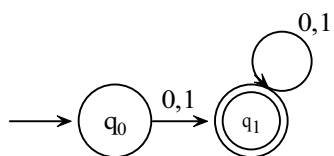
i



l

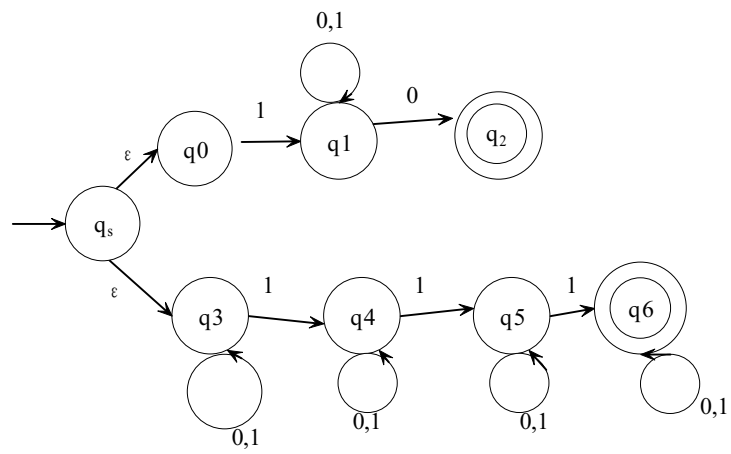


n

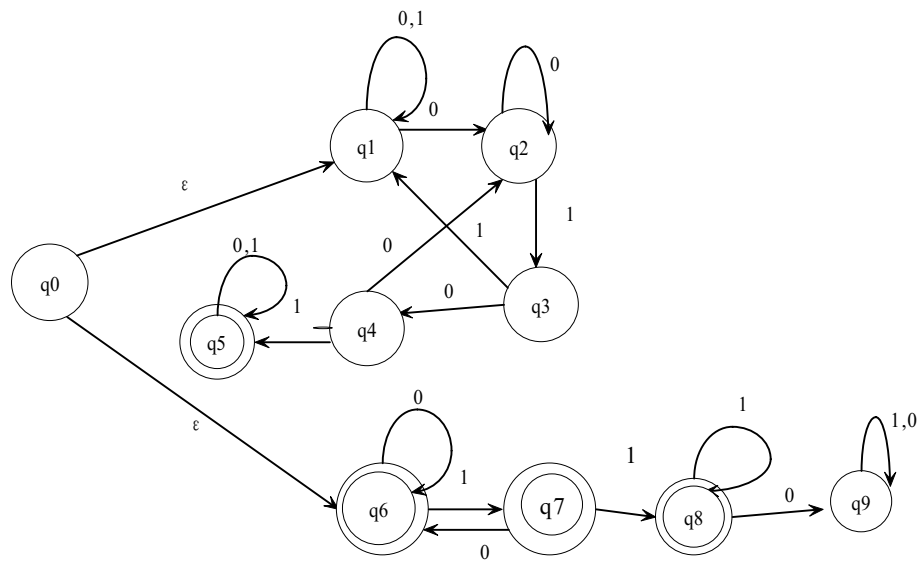


2)1.6.

a.

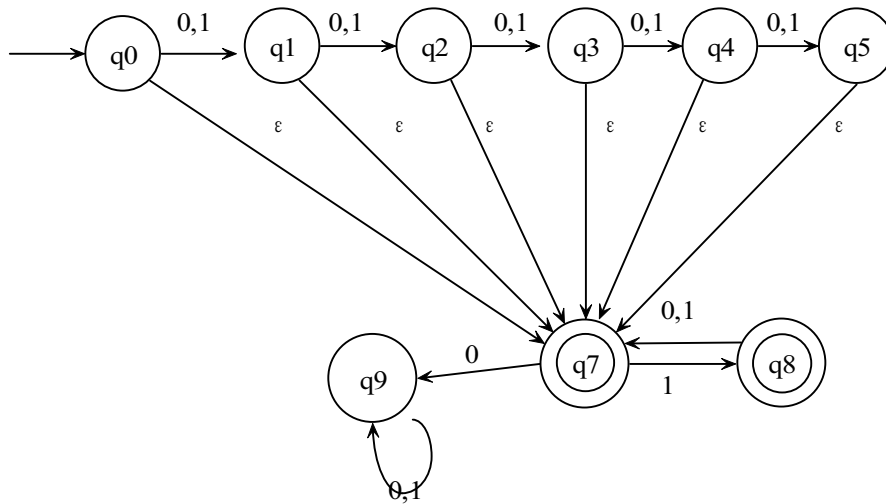


b.

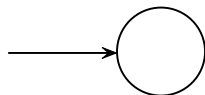


1.7.

a.



b



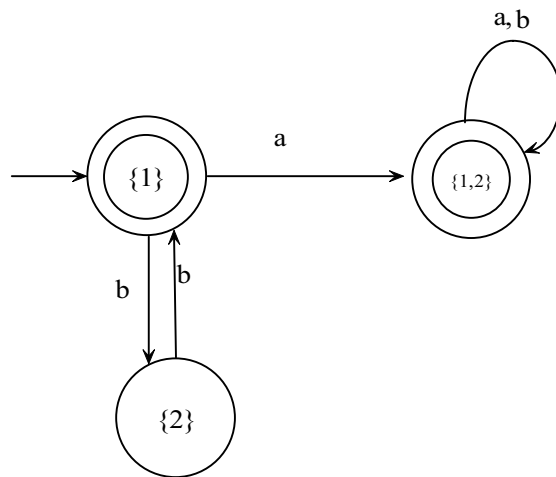
(4) 1.10

- (a) Let M' be the DFA M with the accept and non-accept states swapped. We will show That M' recognizes the complement of B , where B is the language recognized by M . Suppose M' accepts x . Because M and M' have swapped accept/non-accept states, if we run M on x , we would end in a non-accept state. Similarly, if x is not accepted by M' then it would be accepted by M . So M' accepts exactly those strings not accepted by M . Therefore, M' recognizes the complement of B .

Since B could be any arbitrary regular language and our construction shows how to build an automaton to recognize its complement, it follows that the complement of any regular language is also regular. Therefore, the class of regular language is closed under complement.

5) 1.12

(a)



6) 1.13

b. $\Sigma^*1\Sigma^*1\Sigma^*1\Sigma^*$

c. $\Sigma^*0101\Sigma^*$

g. $(\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma)$

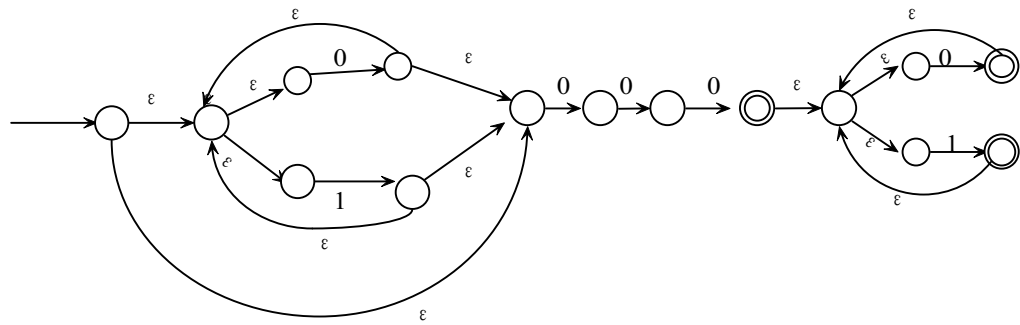
h. $\Sigma^*0\Sigma^* \cup 1111\Sigma^* \cup 1 \cup \varepsilon$

i. $(1\Sigma)^*(1 \cup \varepsilon)$

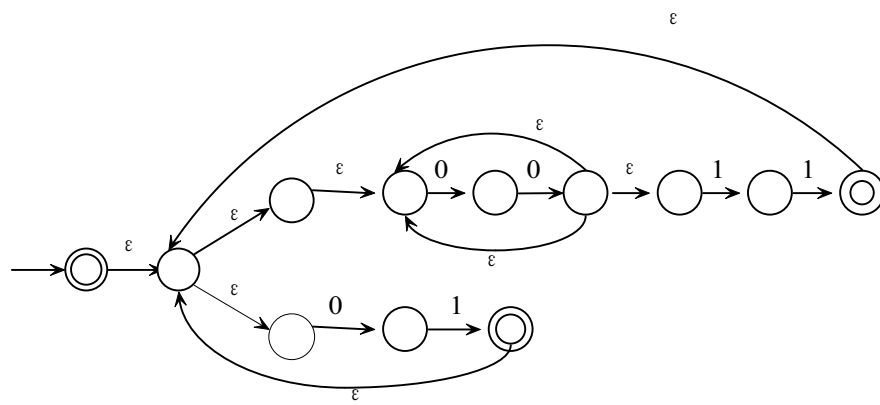
n. $\Sigma\Sigma^*$

7) 1. 14

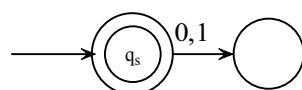
a.



b.



c.



8)1.17

a.

$A_1 = \{0^n 1^n 2^n \mid n \geq 0\}$. Assume that A_1 is regular. Let p be the pumping length given by the pumping lemma. Choose s to be the string $0^p 1^p 2^p$. Because s is a member of A_1 and s has length more than p , the pumping lemma guarantees that s can be split into three pieces, $s = xyz$, where for any $i \geq 0$ the string $xy^i z$ is in A_1 .

- i) The string y consists only of 0s, only of 1s or only of 2s. In the cases the string $xyyz$ will not have equal number of 0s, 1s and 2s, leading to a contradiction.
- ii) 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 it is not a member of A_1 , which is a contradiction.

Therefore, A_1 is not regular.

- b. $A_2 = \{ww \mid w \in \{a,b\}^*\}$. Assume that A_2 is regular. Let p be the pumping length given by the pumping lemma. Choose s to be the string $a^p b a^p b$. The pumping lemma guarantees that s can be split into three pieces, $s = xyz$, where for any $i \geq 0$ the string $xy^i z$ is in A_2 , $|y| > 0$, and $|xy| \leq p$. The only possible case is $y = a \dots a$ and y is to the left of the first b (this is because $|xy| \leq p$). Then $xyyz \notin A_2$.

9) 1.24

For any regular language A , let M_1 be the DFA recognizing it. We need to find a DFA that Recognizes A^R . Since any NFA can be converted to an equivalent DFA, it suffices to find An NFA M_2 that recognizes A^R .

We keep all the states in M_1 and reverse the direction of all the arrows in M_1 . We set the accept state of M_2 to be the start state in M_1 . Also, we introduce a new state q_0 as the start state for M_2 which goes to every accept state in M_1 by an ϵ -transition.

For every string s accepted by M_1 , the path that s follows in M_1 starts at the start state and stops at one of the accept states, say, q_{end} . If we input s^R to M_2 , it starts at q_0 and forks into different paths starting at each of the accept states in M_1 . The computation can reach the accept state in M_2 by following the same path as its reversed counterpart in M_1 but in the opposite direction.

Similarly, for every string w accepted by M_2 , there exists a path P starting at q_0 going through q_{end} as its next step and stopping at the unique accept state. Converting the NFA M_2 back to its counterpart M_1 and reversing the input string, we can get from the start state to q_{end} in M_1 by following the same path in the opposite direction.

Therefore, M_2 recognizes A^R if M_1 recognizes A .

10) (a)

$\{w \mid w \text{ ends with } 11 \text{ or contains at least one } 0\}$

(b)

$\{w \mid w \text{ contains the substring } 111\}$.

Additional Problems:

1.10(b)

Consider the NFA in exercise 1.12(a). The string a is accepted by this automaton. If we swap the accept and reject states, the string a is still accepted. This shows that swapping the accept and non-accept states of an NFA doesn't necessarily yield a new NFA recognizing the complement of the original one. The class of languages recognized by NFAs is, however, closed under complement. This follows from the fact that the class of languages recognized by NFAs is precisely the class of languages recognized by DFAs which we know is closed under complement from part (a).

1.31

Proof Idea: Show that for each all-paths-NFA there exists an equivalent DFA.

Firstly, from the definition of APNFA we can see a DFA is also an APNFA, that is

$L(\text{DFA}) \subset L(\text{APNFA})$.

Next we will show by construction that for each APNFA there is a corresponding DFA.

Let $N=(Q, \Sigma, \delta, q_0, F)$ be the APNFA recognizing some language A . We construct a DFA M recognizing A .

Construct $M=(Q', \Sigma, \delta', q_0', F)$

1. $Q'=P(Q)$.

Every state of M is a set of states of N .

2. For $R \in Q'$ and $a \in \Sigma$ let $\delta'(R,a)=\{q \in Q \mid q \in E(\delta(r,a)) \text{ for some } r \in R\}$

If R is a state of M , it is also a set of states of N .

Recall $E(R)=\{q \mid q \text{ can be reached from } R \text{ by traveling along 0 or more } \epsilon \text{ arrows}\}$.

3. $q_0'=E(\{q_0\})$.

q_0 and all the states that can be reached from the start state of N along the ϵ arrows.

4. $F'=\{R \in Q' \mid R \text{ contains only accept states of } N\}$

States contained in the set R must be **final states** in the APNFA

Clearly, every path for a string in the APNFA can still be followed in the DFA. It is just the case that each state is grouped together with others in the DFA. Every path that used to lead to a final state still leads to a composite state containing that final state. That means that those strings accepted by the APNFA will have paths that lead to a composite state containing only final states, which is also the accept state of the DFA. So the string will be accepted. All other strings lead to different non-accept states, will be rejected.

Finally we get the conclusion that for each APNFA there is an equivalent DFA.

Since the equivalent DFA only recognize the class of regular language then the APNFA recognize and only recognize the class of regular languages.

1.41

The language D is accepted by the following DFA so it is regular.

