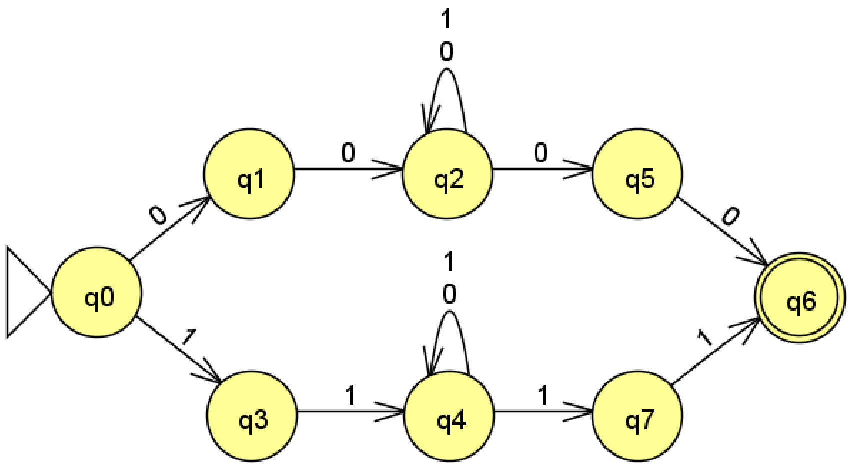
Chapter 3

Non-Deterministic Finite Automata

A. Construct a non-deterministic finite automaton (NFA) for each of the following (assuming that the shortest acceptable strings are long enough to accommodate all the requirements):

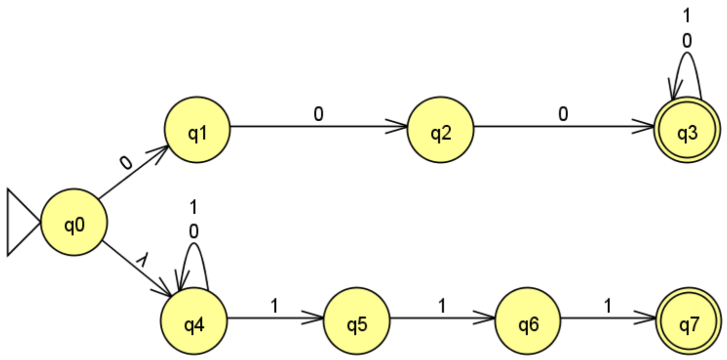
1. Binary strings that begin with 11 and end with 11 or begin with 00 and end with 00.

Solution:



2. Binary strings starting with 000 or ending with 111 (or both).

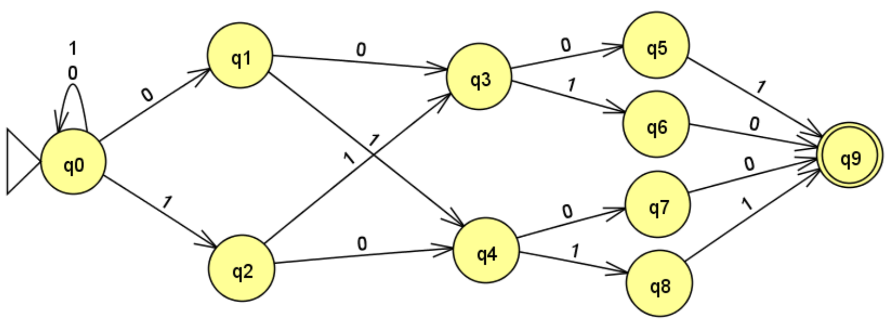
Solution: Strings beginning with 000 are accepted via q_0-q_1-q_2-q_3. Strings ending with 111 are accepted via q_0-q_4-q_5-q_6-q_7. Strings that begin with 000 and end with 111 are accepted in either branch. It is important however that the loops for 0 and 1 on state q_4 are not placed on state q_0 itself.



Please see the VIDEO SOLUTION for this exercise to see how the NFA is constructed.

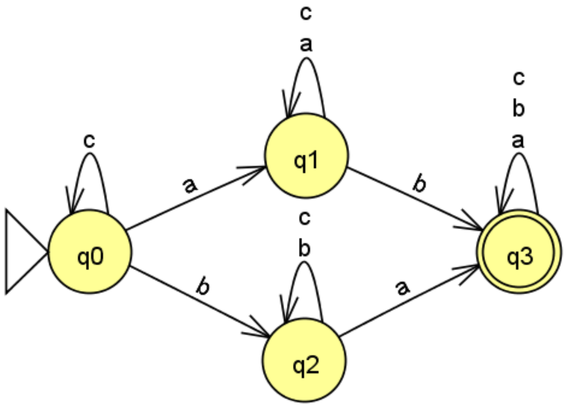
3. Binary strings in which the sum of the last four digits is odd (e.g., 00101011 but not 00101001).

Solution: The sum is odd in q_2, q_4, q_6, q_7 and q_9. It is even in q_0, q_1, q_3, q_5 and q_8.



4. Strings over {*a, b, c*} that contain at least one *a* and at least one *b*.

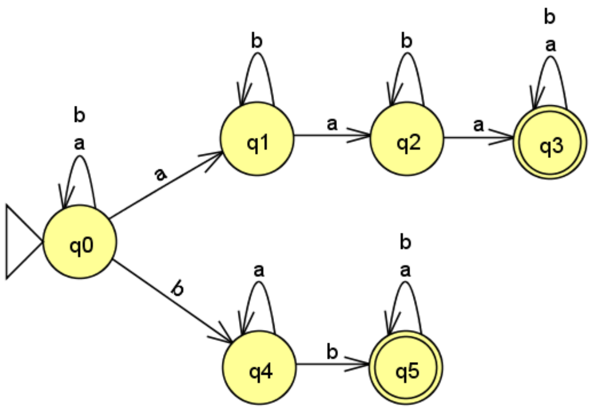
Solution: Note that either the *a* can come first in the input or the *b*. In fact, this automaton is a DFA. It looks for the first *a* and the first *b* (in either order).



Please see the VIDEO SOLUTION for this exercise which shows an NFA instead of the above DFA. Both are, of course, equivalent.

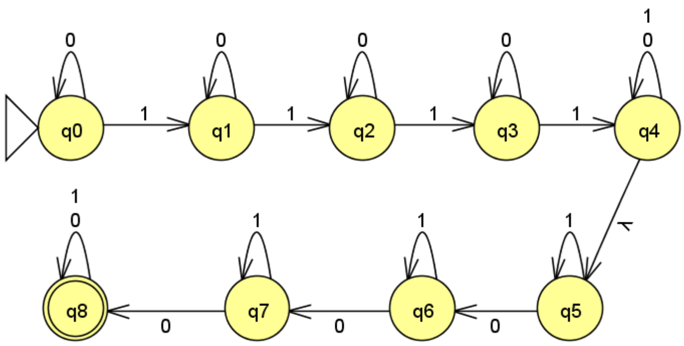
5. Strings over {*a, b*} that contain at least three *a* s or at least two *b* s.

Solution: States in the upper branch count the number of *a* s seen so far and states in the bottom branch count the number of *b* s.



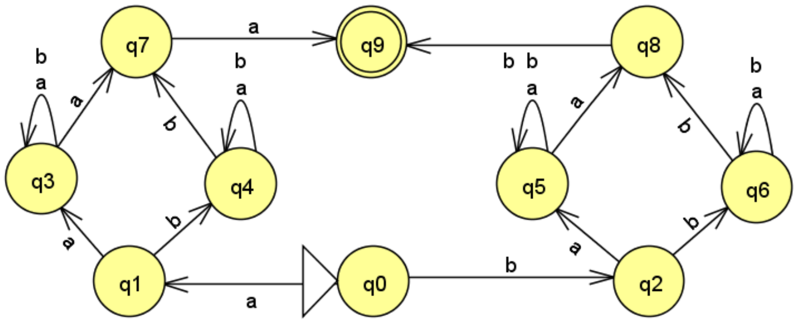
6. Binary strings in which the first part of each string contains at least four 1 s and the second part contains at least three 0 s.

Solution: Note that it is impossible for a finite automaton (DFA or NFA) to find the midpoint of the string to divide it into two equal halves. The two parts can be of unequal length here. Having found the four 1 s in the first part, the NFA jumps to the second half nondeterministically (q_4 to q_5) and looks for three 0 s.



7. Strings over {*a*, *b*} where the last two symbols in each string is a reversal of the first two symbols (i.e., last symbol = first symbol and penultimate symbol = second symbol). The NFA must contain only 10 states (not including reject states).

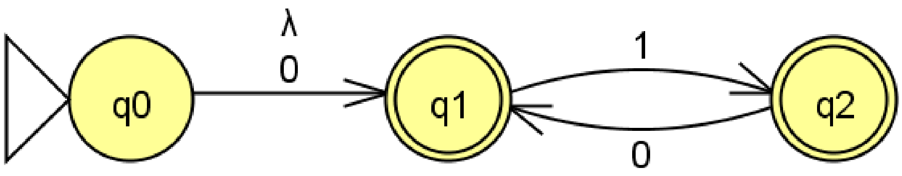
Solution:



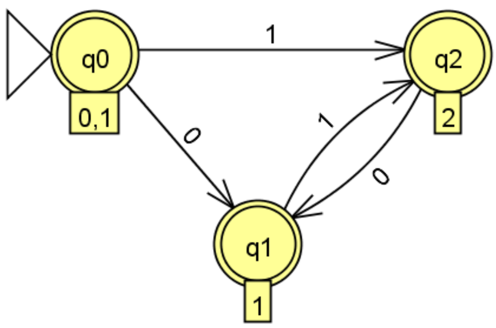
Please also see the VIDEO SOLUTION for this exercise.

8. Binary strings of any length with alternating 0 s and 1 s. The NFA must have just three states (not including reject states). How many states does an equivalent minimal DFA have?

Solution: The λ–transition is needed to accept not only the null string but also all strings beginning with a 1.



An equivalent DFA also has three states (numbers of corresponding NFA states are shown in boxes below the DFA states):

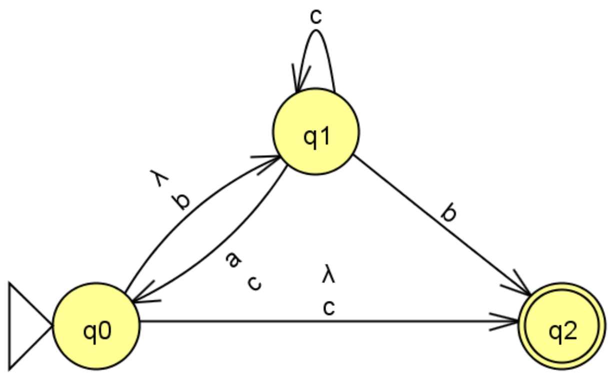


B. For the following problems, a transition table is specified. Convert the given table to an NFA diagram and then to an equivalent DFA.

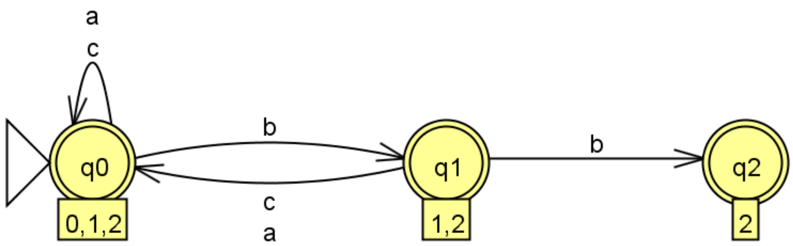
9. The NFA specified by:

State	**Input = *a***	**Input = *b***	**Input = *c***	**λ**
$\rightarrow q_0$	{}	$\{q_1\}$	$\{q_2\}$	$\{q_1, q_2\}$
q_1	$\{q_0\}$	$\{q_2\}$	$\{q_0, q_1\}$	{}
$* \; q_2$	{}	{}	{}	{}

Solution: NFA:



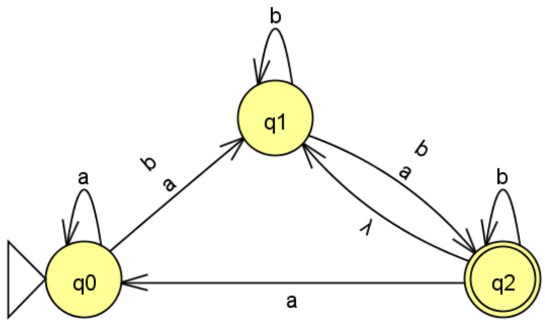
Equivalent DFA:



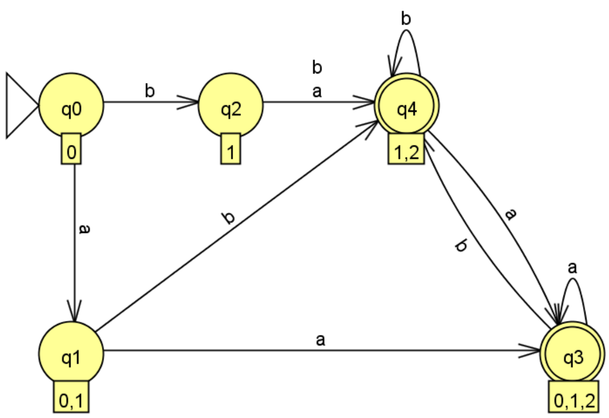
10. The NFA specified by:

State	Input = a	Input = b	Λ
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_1\}$	{}
q_1	$\{q_2\}$	$\{q_1, q_2\}$	{}
$* \; q_2$	$\{q_0\}$	$\{q_2\}$	$\{q_1\}$

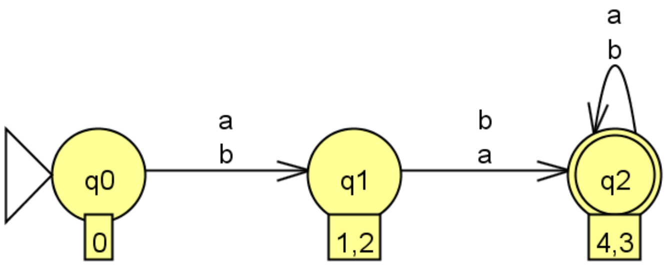
Solution: NFA:



Equivalent DFA:



This DFA can be minimized to obtain the following (where the numbers in the boxes below the states indicate the states of the non-minimal DFA):



C. Convert the following NFAs to equivalent DFAs (and minimize the resulting DFAs):

11. The NFA shown in Figure-3.25.

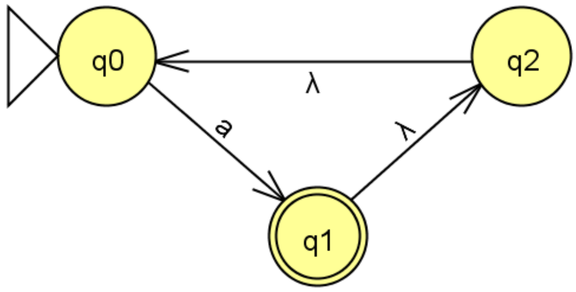
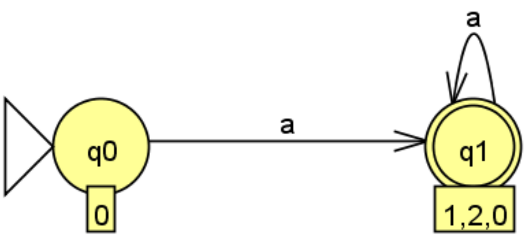


Figure-3.25.

Solution: The DFA obtained from the NFA is already minimal:



12. The NFA shown in Figure-3.26.

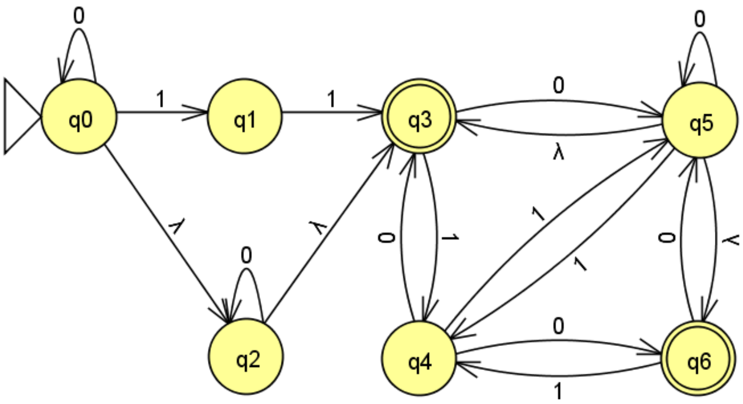
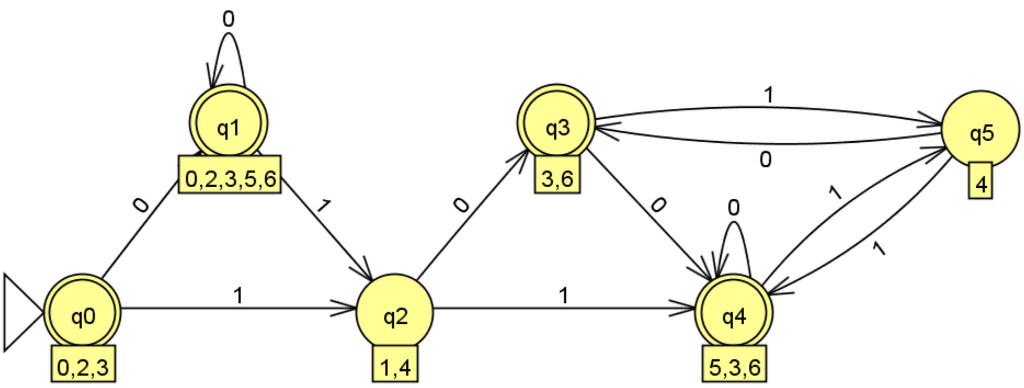
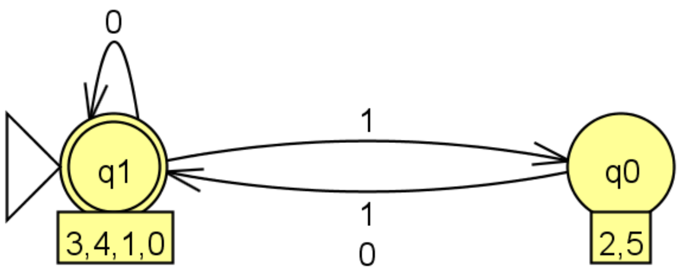


Figure-3.26

Solution: An equivalent DFA is:



This DFA can be minimized to obtain the following DFA with just two states:



13. The NFA shown in Figure-3.27.

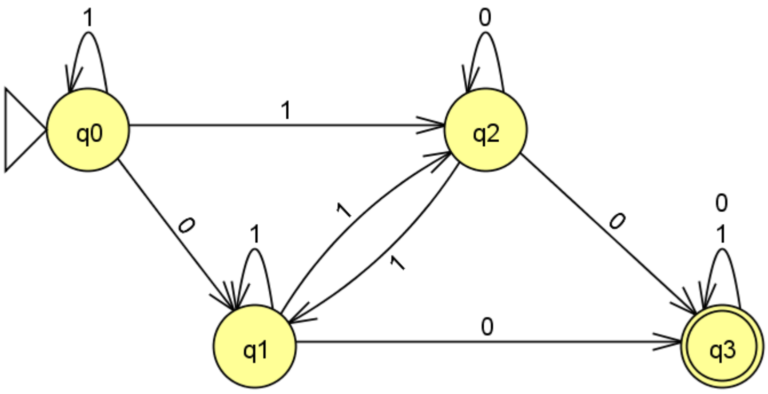
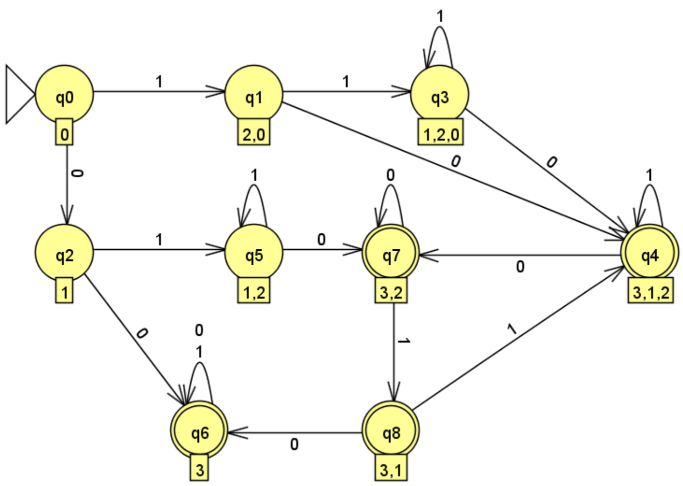
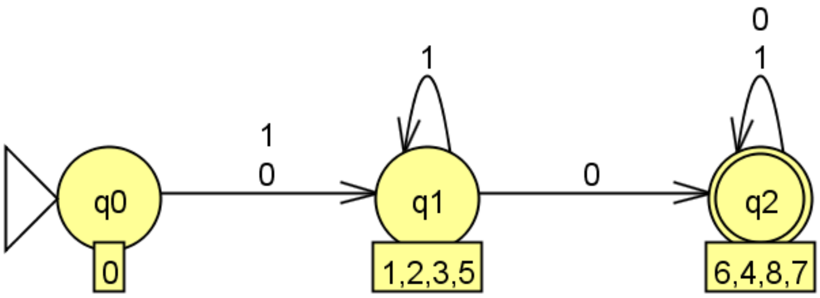


Figure-3.27

Solution: An equivalent DFA is:



After minimizing it, we get:



D. Minimize the following DFAs.

14. The DFA shown in Figure-3.28.

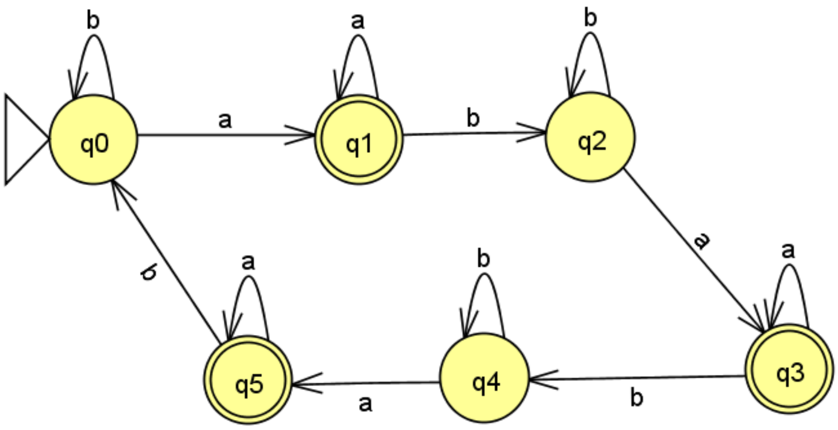
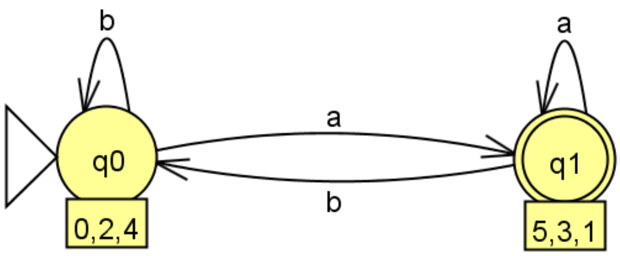


Figure-3.28

Solution: The minimal DFA is:



15. The DFA shown In Figure- 3.29.

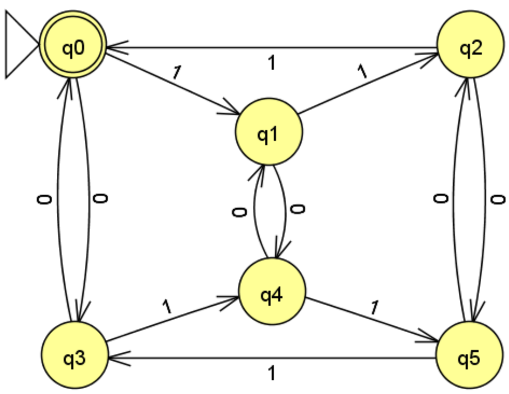


Figure-3.29

Solution: This DFA is already minimal and cannot be minimized further.

16. The DFA shown in Figure-3.30.

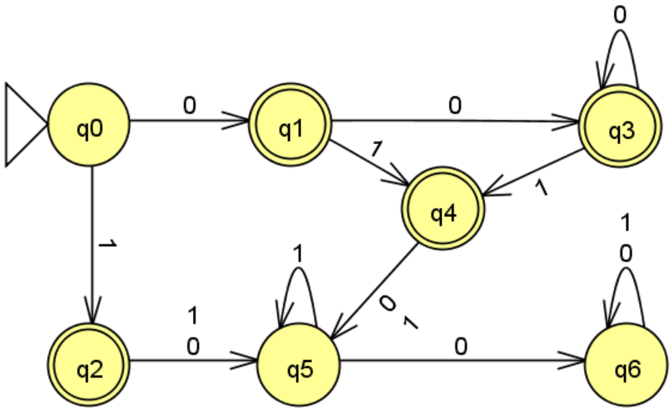
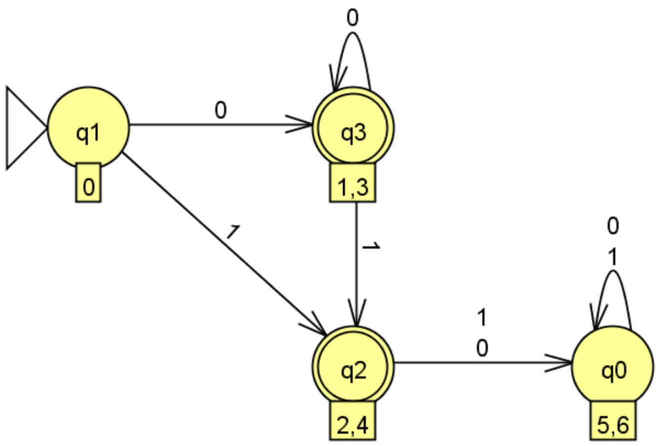


Figure-3.30

Solution: The minimal DFA is:



17. The DFA shown in Figure- 3.31.

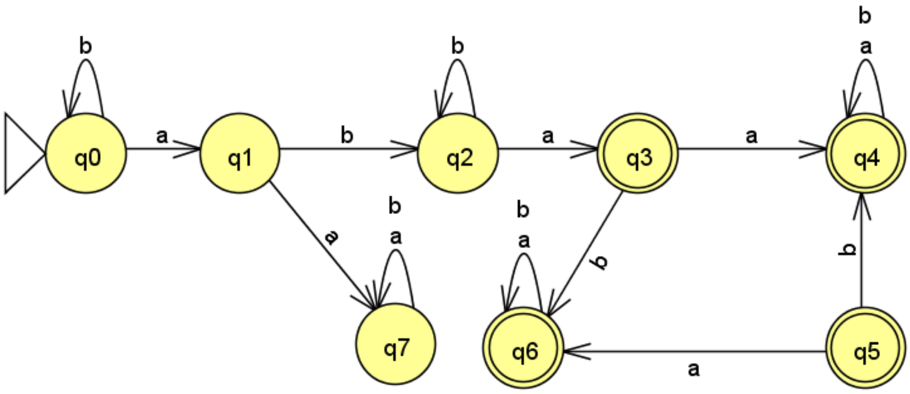
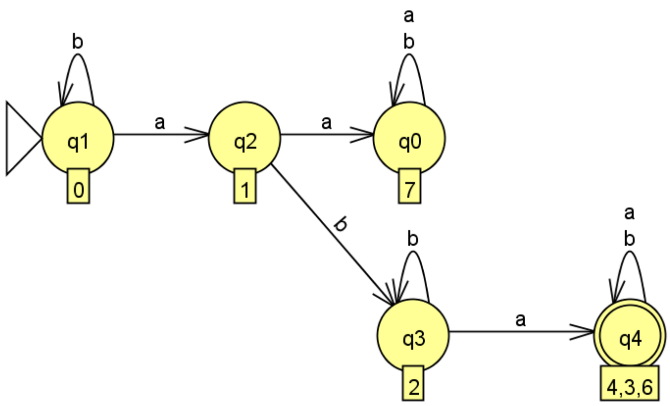


Figure-3.31

Solution: Note that q_5 is unreachable. The minimal DFA is:

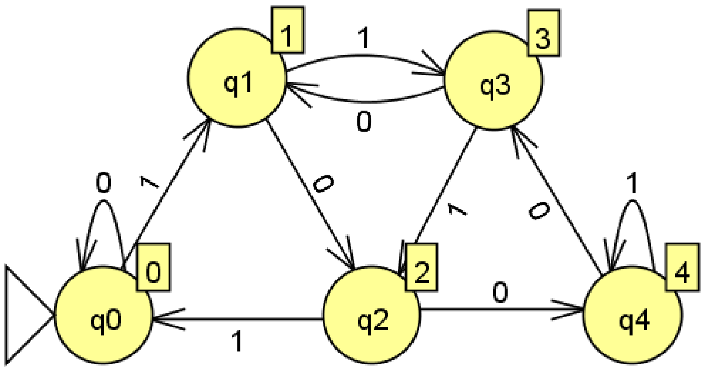


18. What happens if we apply the method of subset construction to a DFA instead of an NFA?

Answer: Nothing happens; the automaton does not change. Every subset will have just one state. We get back the same DFA.

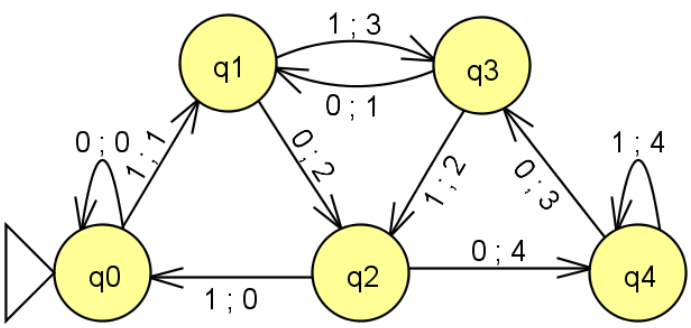
19. Design a Moore machine that takes a binary string as input and outputs after each symbol the remainder obtained when the input thus far is divided by 5 (treating it as a non-negative binary number). The output alphabet is {0, 1, 2, 3, 4}.

Solution:



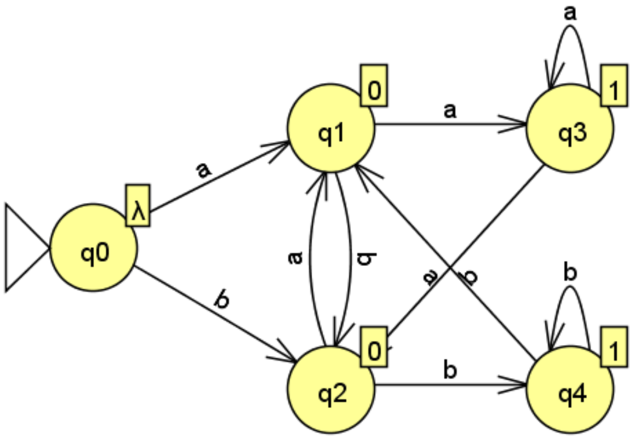
20. Design a Mealy machine to do the same (as in Exercise 19).

Solution:



21. Design a Moore machine to detect a run in the input, that is, sequences of two or more identical symbols. For example, given the input *abaabbbabaa*, the output shall be 00010110001.

Solution:



22. Design a Mealy machine to do the same (as in Exercise 21). }.

Solution: Please also see the VIDEO SOLUTION available for this exercise.