Solution 1:

Correct Answer: (D)

As instructor discussed in lecture 5, that any NFA can have multiple computation path for an input string \( w \). Let \( Q' \) be the set of states, where all the computation path ends up after reading last symbol of \( w \). Now if there exist a state \( q \in Q' \) such that \( q \) is a final state in NFA then we say NFA accepts \( w \).

Solution 2:

Correct Answer: (C)

As given in the question, Let \( A \) and \( B \) be two finite languages with cardinality (number of strings in the language) \( p \) and \( q \) respectively. Assume \( C = A.B \), i.e. \( C = \{ w_1w_2 \mid w_1 \in A \text{ and } w_1 \in B \} \). Since \( w_1 \in A \) and \( w_2 \in B \) there are \( p \) choices for \( w_1 \) and \( q \) choices for \( w_2 \). So total number of possible string \( w \in C \) are \( pq \). Similarly let \( C = A \cup B \) i.e \( C = \{ w \mid w \in A \text{ or } w \in B \} \). We know that by standard set theory \( |C| \leq |A| + |B| \). Hence \( |C| \leq p + q \).

Solution 3:

Correct Answer: (C)

Explanation: We will only prove that \( L_2 \) is also a language of the given automaton as \( L_1 \) was explained as an example in lecture-3. Lets name the automaton given in question as \( A \), lets swap the start and final state and reverse all the edges of \( A \) and call that automaton \( A' \), one can easily observe that \( A = A' \). Clearly if a string \( w \) was accepted in \( A \) then its reverse \( w^R \) will be accepted by \( A' \) and vice versa because we have reversed all the edges and swapped the final state and initial state of \( A \) to get \( A' \).

One other way of proving that \( L_2 \) is the language of \( A \) is to prove that a binary string is divisible by 3 iff its reversal is divisible by 3. We are leaving that as an exercise.

Exercise: Let \( A \) be DFA for the set of binary strings which are divisible by odd number \( k \), and \( A' \) be the automaton by reversing all edges and swapping initial and final state. Then prove that the language of \( A' \) is the set of binary strings whose reversal is divisible by \( k \).

Solution 4:

Correct Answer: (B)

Explanation: We will cover DFA minimization more formally later in the course but one can easily argue that why DFA for set of words which contain some substring \( w \) can not have less than \( |w| + 1 \) states. The main idea is that since we are looking for the presence of the word \( w \) therefore we need to store the information for last \( |w| \) characters at every point of time while processing the input. So in total it will have at least \( |w| \) states for storing the information and one initial state. So its proven that the automaton for the set of binary words with four consecutive 1s will have at least 5 states, but to show that indeed 5 is the minimum number we are giving the below the automaton for the language.
Solution 5:

Correct Answer: (D)
Explanation: Option 1) is incorrect because automaton is not accepting 1001. Option 2) is incorrect because automaton is not accepting 1101. Option 3) is incorrect because automaton is accepting 11. To see why option 4) is correct let's look at the automaton $A'$ made by swapping accepting and non-accepting state i.e. make $q_3$ the only accepting state and $q_0, q_1$ and $q_2$ non-accepting state, if we can prove that the $A'$ accept the set of all binary strings which contain at least two 1s separated by at least one 0 we'll be done. Clearly the strings accepted in $A'$ have at least two 1s, because to reach $q_3$ we have to go through the edges $(q_0, q_1)$ and $(q_2, q_3)$ and since we can not skip the edge $(q_1, q_2)$ it also ensures that those two 1s will be separated by at least one 0. This proves that string accepted by the $A'$ has at least two 1s separated by at least one 0.

To prove the converse i.e. every string with at least two 1s separated by at least one 0 will be accepted by $A'$, notice that every such string $w$ can be broken down into $xyz$ such that $y$ consists of all 0s, $z$ starts with 1 and $x$ starts with $k_1$ many 0s followed by $k_2$ many 1s where $k_1 \geq 0$ and $k_2 > 0$. Clearly at the end of the $x$ automaton will be at $q_1$ and after processing $y$ it will be at $q_2$ and then due to 1 in the beginning of $z$ it will reach $q_3$ and stay there and hence the string $w$ will be accepted.

Solution 6:

Correct Answer: (C)
Discussed by instructor in lecture 5, From a state in NFA upon reading a symbol from input string, NFA can make transition to at most all states of NFA.

Solution 7:

Correct Answer: (C, D)
Explanation: As explained in lectures, there are infinitely many DFA which accepts a given regular language. Hence option D is correct. Also, the transition diagram of a DFA can also be interpreted as an NFA; hence C is also correct. The other options contradict D and hence are wrong.

Solution 8:

Correct Answer: (B)
DFA is very much similar to the NFA except their transition function. In DFA from an state, on reading any symbol we can go to exactly one state. But in NFA we can go to more than one state or none at all. So we can think of DFA an special case of NFA, where from every state there is exactly one transition on reading a particular symbol. So we can say that every DFA is an NFA but not vice versa. Hence (B) option is correct.

Solution 9:

Correct Answer: (D)
Let $N_1$ be the NFA given in question and $N_2$ the NFA obtained by removing the following two transition, $\delta(q_0, a) = q_1$ and $\delta(q_1, b) = q_3$. Notice that removing these two transition did not affect the language accepted by the NFA at all i.e. $L(N_1) = L(N_2)$. Because in $N_1$ if we choose the computation path (say $p_1$) "$q_0, a, q_1, b, q_3$" to reach $q_3$ from $q_1$. In this path we are reading only additional substring "ab" to reach $q_3$. Now in $N_2$ to read substring "ab" we can choose computation path $q_0, a, q_0, b, q_3$. Hence having $p_1$ does not provide any addition substring which we can not have in $N_2$ from moving state $q_1$ to $q_3$. Hence both accept the same language. Now it is very easy to find the language accepted by $N_2$. It is $(a + b)^*bb(a + b)^*$. First $(a + b)^*$ because of self loop in $q_0$ then we can go to $q_1$ to $q_2$ by reading only substring "bb" and then $(a + b)^*$ from loop in $q_4$. It is clear that $(a + b)^*bb(a + b)^*$ is the regular expression form set of string which contains $bb$ as a substring.
Solution 10:

Correct Answer: (A),(B),(C)
Option (A) is not correct because let \( w \in A \cdot B = \{10, 11\} \), such that \( w_1 \in A \) and \( w_2 \in B \). Integer denoted by \( w \) is 11 which is not divisible by 6. Option (B) is also not correct let \( w \in B \cdot A = \{11, 10\} \), such that \( w_1 \in B \) and \( w_2 \in A \). Integer denoted by \( w \) is 14 which is not divisible by 6. From above two we can conclude option (C) is also not correct. Option (D) is correct. Choose \( A = B = \{\epsilon\} \).

Solution 11:

Correct Answer: (B)
Option (A) is not correct because DFA accepts \( \epsilon \) string, which does not contains any \( b \)'s at all. Option (C) is also not correct since DFA accept string \( b \) which is not ending with an \( a \). Option (C) is correct because whenever i see an \( a \) in input string i reach to state \( q_1 \) but at \( q_1 \) on reading another \( a \), computations reach to dump state. So DFA cannot accept a string which contains \( aa \) as a substring. On other hand i can see any number of \( b \)'s before and after an \( a \). Hence (B) is correct.

Solution 12:

Correct Answer: (A, D)
Explanation: The given language \( \Sigma^2 \) consists of strings of length 2 whose alphabets are from the set \( \Sigma \). The requirement is only satisfied by the options A and D.