**CS21004 Formal Languages and Automata Theory, Spring 2011–12**

**Class test 1**

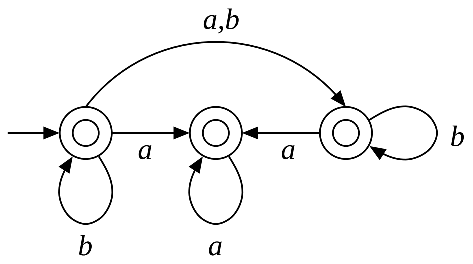
Maximum marks: 20 | Date: 09-February-2012 | Duration: 1 hour

---

**Roll no:** ____________________ **Name:** ____________________________________________

[ *Write your answers in the question paper itself. Be brief and precise. Answer* *all* *questions.* ]

**1.** Consider the following NFA (over the alphabet $\{a, b\}$).



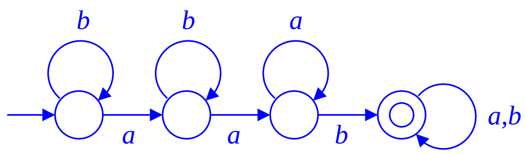

**(a)** What is the shortest string *not* accepted by this NFA? ______ $aab$ ______ **(1)**

**(b)** Let $L_1$ denote the set of all strings *not* accepted by this NFA. Write a regular expression for $L_1$. **(3)**

$b^*ab^*aa^*b(a+b)^*$

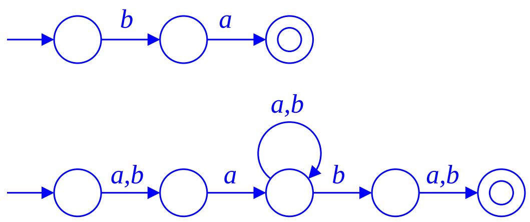
**(c)** Convert the regular expression of Part (b) to an equivalent NFA. **(4)**

*Solution*



**2.** Let $L_2$ denote the language over the alphabet $\{a, b\}$ such that $\alpha \in L_2$ if and only if $|\alpha| \geqslant 2$, the second symbol of $\alpha$ is $a$, and the second last symbol of $\alpha$ is $b$. Design an NFA to accept $L_2$. **(6)**

*Solution*




**3.** Let $\alpha$ be a string (over some alphabet $\Sigma$). By $\operatorname{odd}(\alpha)$, we refer to the string obtained by deleting symbols at all even positions of $\alpha$. That is, if $\alpha = a_1a_2a_3 \ldots a_n$, then $\operatorname{odd}(\alpha) = a_1a_3a_5 \ldots a_{n'}$, where $n'$ is $n$ or $n-1$ according as whether $n$ is odd or even. For a language $L \subseteq \Sigma^*$, define $\operatorname{odd}(L) = \{\operatorname{odd}(\alpha) \mid \alpha \in L\}$. Prove that if $L$ is regular, then $\operatorname{odd}(L)$ is regular too. **(6)**

*Solution* Let $M = (Q, \Sigma, \delta, s, F)$ be a DFA accepting $L$. We design an $\epsilon$-NFA $M' = (Q', \Sigma, \delta', S', F')$ to accept $\operatorname{odd}(L)$. Let $Q = \{p_1, p_2, \ldots, p_k\}$. Then, we take $Q' = \{p_1, p_2, \ldots, p_k, q_1, q_2, \ldots, q_k\}$, that is, we add a new state $q_i$ for each $p_i$. The only start state of $M'$ is the old start state $s$ of $M$, that is, $S' = \{s\}$. All the final states of $M$ continue to remain final in $M'$. Moreover, a new state $q_i$ is marked final in $M'$ if and only if the corresponding old state $p_i$ is final (in $M$). We *replace* each transition $p_i \xrightarrow{a} p_j$ in $M$ (where $a \in \Sigma$, and where we may have $i = j$) by the two transitions $p_i \xrightarrow{a} q_j$ and $q_i \xrightarrow{\epsilon} p_j$.

By construction, moves in $M'$ alternate between the old and the new states. In essence, $M'$ mimics $M$ with the only exception that every second move prevents consuming a real symbol from the input.