Reg. No. .....

Indian Institute of Information Technology Surat
Department of Computer Science and Engineering
Mid Semester Examination, 3rd Sem, 2025-26
Automata and Formal Languages (CS304)

Timing: 3:30PM to 5:00PM **Date:** 29 Sep 2025 **Maximum Marks:** 30

Read all instructions: Ensure that your solutions follow all the conventions discussed in the class. Attempt all parts of a question at one place. If you cancel some part of your solution, please strike it out clearly. Last section contains optional questions of increasing difficulty. You must attempt only one. Answering a more challenging question correctly will result in a higher total score for the exam. Choose wisely.

Question 1. *Prove using induction that for all $n \geq 2025$, the number $4^{2n+1} + 3^{n+2}$ is divisible by 13. Clearly state the base case, induction hypothesis, and inductive step.* ***(5)***

Question 2. *State whether the following are True or False. Justify your answer with a brief explanation or a counterexample. Attempt* ***any 5*** *of the following. If you attempt more than 5, only the first 5 will be considered for evaluation.*

a. *For the language $L = \emptyset$, $L^* = L$.* ***(2)***

b. *Irregular languages are closed under complementation.* ***(2)***

c. *Let DFA_1 be the class of DFAs with exactly one accept state. Then, DFA_1 is less powerful than DFA.* ***(2)***

d. *Let $NFA_{no\text{-}\varepsilon}$ be the class of NFAs without any ε-transition. Then, $NFA_{no\text{-}\varepsilon}$ is less powerful than NFA.* ***(2)***

e. *In the state-elimination algorithm to convert NFA to RE, the order in which the states are eliminated matters. In other words, different orders of state elimination can lead to different regular expressions describing distinct regular languages.* ***(2)***

f. *There exists a pumpable irregular language.* ***(2)***

g. *The language of all valid C programs is regular.* ***(2)***

Question 3. *Consider the alphabet $\Sigma = \{a, b\}$. Let L_1 be the language defined by the regular expression $R_1 = (ab)^*$. Let L_2 be the language defined by NFA N_2 as follows:*

- *States: $Q = \{q_0, q_1, q_2\}$ where q_0 is the start state and q_2 is the only final state;*
- *Transitions: $\delta(q_0, b) = \{q_1\}$, $\delta(q_1, b) = \{q_1, q_2\}$, and $\delta(q_2, a) = q_0$.*

Answer the following ***without any proofs****. If you do not know how to solve a part, you can use still use the solution for that as a black-box to answer the rest of the parts.*

a. Construct a diagram for an NFA that accepts L_1. ***(1)***

b. Construct a diagram for the NFA N_2. ***(1)***

c. Construct a diagram for an NFA that accepts L_1L_2. ***(1)***

d. Write a regular expression for L_2. ***(1)***

e. Write a regular expression for L_1L_2. ***(1)***

Designing Unambiguous CFGs.

For this section, consider the alphabet $\Sigma = \{a, b, c, >, +, =\}$. Attempt **any one question** from Questions 4-6. If you attempt more than one question, only the first one will be considered for evaluation. State your answers **without any proofs**. Note that they have increasing maximum scores. For maximum score, attempt Question 6.

Question 4. *Let L_1 be the language of C-style addition expressions over $\{a, b, c\}$. Thus, L_1 contains strings denoting expressions like $a + b$ and $b + a + c + a$ but not ones like $+a$ or $a+$ or $a = b$ or $ab + c$. Design an unambiguous CFG for L_1 that breaks ambiguity by evaluating all addition operators from left to right. Draw a parse tree for $a + b + c$ to show that it is parsed as $(a + b) + c$.* ***(5)***

Question 5. *Let L_2 be the language of C-style expressions over $\{a, b, c\}$ using operators $+$ (addition), $>$ (strictly greater than), and $==$ (equality). Thus, L_2 contains strings denoting expressions like $a > b + c == a$ and $a + b > b$ but not ones like $a+$ or $a = b$ or $ab + c$. Design an unambiguous CFG for L_2 that breaks ambiguity as follows:*

- *all addition operators are evaluated first, in left to right order; and*
- *all relational operators are evaluated next, in left to right order.*

Draw parse tree for $a > b + c == a$ to show that it is parsed as $(a > (b + c)) == a$. ***(7)***

Question 6. *Let L_3 be the language of C-style expressions over $\{a, b, c\}$ using operators $+$ (addition), $>$ (strictly greater than), $>=$ (greater than or equal to), $==$ (equality), and $=$ (assignment).* [1] *Thus, L_3 contains strings denoting expressions like $a = b = c$ and $a = b == c$ and $b = a > b + c >= a$, but not ones like $a == b = c$ or $ab + c$. Design an unambiguous CFG for L_3 that breaks ambiguity as follows:*

- *all addition operators are evaluated first, in left to right order;*
- *all relational operators are evaluated next, in left to right order; and*
- *all assignment operators are evaluated last, in right to left order.*

Draw parse trees for $a > b + c >= a$ and $a = b = c$ to show that they are parsed as $(a > (b + c)) >= a$ and $a = (b = c)$ respectively. ***(10)***

[1] Note that there is no $+=$ (compound assignment) operator for L_3.