25

B.E/B.Tech.(Full-time) Degree End Semester Examinations, April/May 2011
Branch : Information Technology
IV Semester
IT9251 – Formal Languages and Automata - R-2008

Time: 3 Hours **Max. Marks: 100**

Answer all questions
PART – A (10 x 2 = 20 Marks)

1. Define DFA and NFA. Write the conventions used in providing meanings to symbols and the symbols used in Finite Automation.

2. Prove that $1^2 + 2^2 + 3^2 + n^2 = \frac{n(n+1)(2n+1)}{6}$ using induction.

3. What is meant by a regular expression? Write a regular expression to denote the variable names in a programming language which accepts strings starting with a letter followed by either letter or digit any number of times.

4. Prove that regular sets are closed under union and intersection.

5. What is an ambiguous grammar? Give an example.

6. Define Push Down Automata. State its applications.

7. How will you eliminate ϵ – Productions? Explain it using the grammar
   $S \rightarrow aA$
   $A \rightarrow BB$
   $B \rightarrow aBb / \varepsilon$

8. Define Turing machine. Explain it briefly.

9. What is an LL (1) Grammar? Give an example.

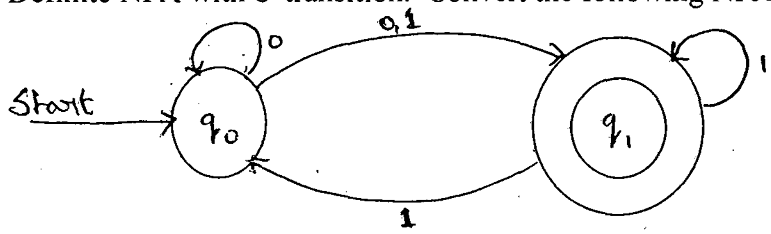
10. Define LR (0) and LR (k) grammar. State their applications.

PART – B (5 x 16 = 80 Marks)

11. (i) For any positive integers i, j and n, if i *j = n, then prove that either $i \leq \sqrt{n}$ or $j \leq \sqrt{n}$. (6)

(ii) Let L be a set accepted by a nondeterministic finite automation. Then prove that there exists a deterministic finite automation that accepts L. (6)

(iii) Definite NFA with ε–transition. Convert the following NFA into DFA. (4)



12. (a) (i) State and prove the pumping lemma for regular sets. Write the applications of pumping lemma. (8)

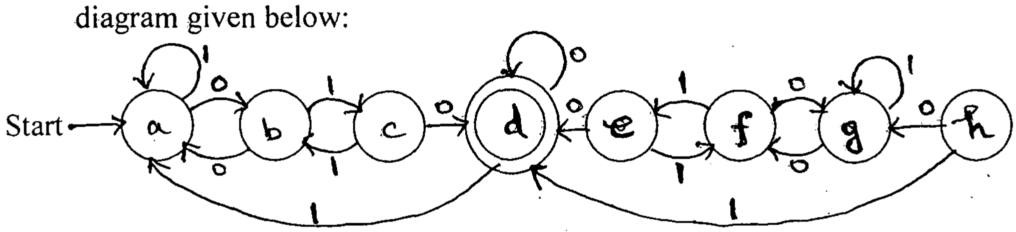
(ii) Using Pumping lemma, prove that the language $\{a^n b^n \mid n \geq 0\}$ is not regular (4)

(iii) Write the closure properties of regular languages. (4)

OR

(b) (i) Let r be a regular expression describing the language L. Then prove that there exists an NFA with ε – transition that accepts L. (8)

(ii) Find the minimum-state finite automation equivalent to the transition diagram given below: (8)



13.(a) (i) Define context free grammar, parse tree, left most derivation and right most derivation. (4)

(ii) You are given the following unambiguous grammar for expressions. (4)

<assign> ⟶ <id> : = <expr>
<id> ⟶ A|B|C
<expr> ⟶ <expr> + <term>
| <term>
<term> ⟶ <term> * <factor>
| <factor>
<factor> ⟶ (<expr>)
| <id>

Using this grammar, illustrate the left most and right most derivation for the expression A:=B + C *A.

(iii) What is left recursion? How will you eliminate left recursion? Eliminate the left recursion from the following grammar: (8)

S ⟶ Aa|b
A ⟶ Ac|Sd|ε

(OR)

13. (b) (i) Let G be a context free grammar having the productions $S \rightarrow aSa|bSb|c$. If G generates the language
$L = \{ xcx^r | x \in \{a,b\}^* \}$
which consists of strings of palindromes, Design a Push Down Automation for L. Check whether "abcba" is a palindrome using this PDA moves. (6)

(ii) Design a deterministic PDA for the language
$L = \{x \in \{a,b\}^* | n_a(X) > n_b(x)\}$.
Illustrate the operation of this machine using the string "abbabaa". (6)

(iii) Construct a PDA equivalent to the grammar
$S \longrightarrow aAA$,
$A \longrightarrow aS|bS|a$
and explain the method used by you. (4)

14. (a) (i) Define Chomsky Normal Form. Write the procedure used to converting a set of productions to CNF. (4)

(ii) Convert the following context-free grammar ({S,A,B}, {a,b}, P, S) to Chomsky Normal form. (4)

(iii) Convert the grammar G given by $G = (\{A_1, A_2, A_3\}, \{a,b\}, P, A_1)$ where P consists of the following productions:
$A_1 \rightarrow A_2A_3$
$A_2 \rightarrow A_3A_1 | b$
$A_3 \rightarrow A_1A_2 | a$
to Greibach Normal Form. Explain the procedure used to convert a context free language to Greibach Normal Form. (8)

OR

(b) (i) State the Pumping Lemma for Context Free Languages and explain the steps used in it. Using this lemma, prove that the language $L = \{ o^n 1^n | n \geq 1\}$ is not a context free language. (6)

(ii) Design a Turning machine to accept palindromes over {a,b}. Check it with examples for odd and even palindromes. (6)

(iii) Design a Turning machine to find whether a given number is prime or not. (4)

15 (a) (i) Write the procedures used to compute FIRST( ) and FOLLOW( ). Using this procedure, find the FIRST( ) and FOLLOW( ) for the non-left recursive grammer.

$E \rightarrow TE'$

$E' \rightarrow + TE' / \varepsilon$

$T \rightarrow FT'$

$T' \rightarrow * FT' / \varepsilon$

$F \rightarrow (E) / id$ (5)

(ii) Write the algorithm used to construct a predictive parsing table. Using this algorithm, construct a predictive parsing table for the non-left recursive grammar given above (in 15a(i)). (5)

(iii) Explain the predictive parsing algorithm. Generate the moves made by a predictive parser on input id + id * id. (6)

OR

(b) (i) Explain the LR-parsing algorithm in detail (8)

(ii) Design a bottom up parser for the context Free Grammar G with productions

$S_1 \rightarrow S_1 + T$

$S_1 \rightarrow T$

$T \rightarrow T * a$

$T \rightarrow a$

and explain the moves for a + a * a $ (8)