Unit 2: Lexical Analysis

Q.1. Explain compiler construction tools with example?

Q.2. what is the role of lexical analysis? Explain input buffering with example?

Q.3. Write the algorithm of lexical analyzer to identify strings, sequences, comments, reserved words and identifiers?

Q.4. Construct minimum state DFA for the regular expression (a|b)*a (a|b)

Q.5. Construct NFA for the regular expression (0 | 1)* 00) |0)

Q.6. Construct DFA for accepting the following language over an alphabet {0,1}
a) Accept only 1 as a string.
b) Accept string as 01.
c) Number of 1's is even and number of 0's is even.
d) Number of 1's is odd and number of 0's is odd.

Q.7. Construct NFA for following regular expression: (using Thompsons rule)
1) (a*|b*)*
2) (a|b)*a(a|b)(a|b)

Q.8. Construct DFA for the following language:
1) All strings starting with 011.
2) All strings starting with 100
3) All strings ending with 011
4) All string with a as a substring i.e. 011 anywhere in the string.

Q.9. Construct NFA for the regular expression (a* | b*)* using Thompson's Rule.

Q.10. Draw a transition diagram to represent relational operators.

Q.11. Construct NFA from a regular Expression (a/b)* abb. Convert it into DFA using subset construction method.

Unit 4: Syntax Directed Translation

1. Write grammar and SDD for following assignment statement x: = a + b * c + d. construct annotated parse tree.
2. What is S-attributed and L-attributed Definition? Implement bottom up evaluation of S-attributed for the input "3 * 5 + 4n".
3. Write grammar and SDD for converting infix expression into postfix.
4. What is three address code? Translate the expression (a+b)/(c+d)*(a+b/c)-d into quadruples, triples and indirect triples.
5. Generate three address code for following Boolean expression (a < b) or (c < d) and (e < f) using the translation scheme of Boolean.
6. What is three address code? What are the types of three address code? Write SDD to generate three address code for assignment?
7. Explain back patching with suitable example?.
8. How would you generate three address code for flow of control statements? Explain with example.

OR

9. **Write the grammar for flow-of-control statements?**
   The following grammar generates the flow-of-control statements, if-then, if-then-else, and while-do statements.
   S -> if E then S1
   | If E then S1 else S2
   | While E do S1.

10. Write the grammar and sematic rule for Boolean expression(OR, AND, NOT)