Reg. No. : | | | | | | | | | | |

Question Paper Code: 36201

B.E. / B.Tech. DEGREE EXAMINATION, APRIL 2019

Sixth Semester

Computer Science and Engineering

01UCS601 - PRINCIPLES OF COMPILER DESIGN

(Regulation 2013)

Duration: Three hours Maximum: 100 Marks

Answer ALL Questions

PART A - (10 x 2 = 20 Marks)

1. Define Lexemes, Tokens, and Patterns.
2. Distinguish between compiler and interpreter.
3. Differentiate between final states in a NFA and a DFA.
4. What does the regular expression (0* /1*)* denote?
5. What is the significance of look-ahead symbols in LR (1) items? When do they loose their significance?
6. Write the drawbacks of shift-reduce parser.
7. Illustrate why every S-attributed definition is L-attributed.
8. What is annotated parse tree?
9. What is flow graph?
10. Write down the different forms of intermediate code.

PART - B (5 x 16 = 80 Marks)

11. (a) What are the phases of compiler? Explain with a neat diagram. Also write down the output for the following expression after each phase a: = b+ c*d/e. (16)

Or

(b) (i) Write a note on language processors. (12)

(ii) Discuss about compiler construction tools. (4)

12. (a) Construct an NFA to recognize the regular expression $(a^+ b \mid a^+)^+$, Obtain its equivalent DFA and minimize the number of states in DFA. (16)

Or

(b) Design a Lexical analyzer generator. Also write the sample code which includes declaration, translation rules and auxiliary procedures. (16)

13. (a) Consider the following context free grammar G = ({S, A, B}, S, {a, b}, P)
where P is

S -> Aa/ bAc/ dc/ bda

A -> d.

Show that this grammar is LALR (1) but not SLR (1). (16)

Or

(b) Consider the following grammar

D → type tlist;

t → double | float

tlist → tlist, id | id

Construct SLR parsing table and find whether string double id, id; is correct or not using the table. (16)

14. (a) Write the syntax directed definition for generating 3-address code for an assignment statement. (16)

Or

(b) Discuss the various storage allocation strategies and their merits and demerits. (16)

15. (a) Explain the common sub expression elimination, copy propagation, and transformation for moving loop invariant computations in detail. (16)

Or

(b) (i) Write down the issues in the design of Code generator. (8)

(ii) Formulate an algorithm for code generation. (8)
