Question Paper Code | **12193**

B.E. / B.Tech. - DEGREE EXAMINATIONS, NOV / DEC 2023
Sixth Semester
Computer Science and Engineering
20CSPC602 - COMPILER DESIGN
(Regulations 2020)

Duration: 3 Hours Max. Marks: 100

PART - A (10 × 2 = 20 Marks)
Answer ALL Questions

Marks,
K-Level, CO

1. Differentiate compiler and interpreter. — *2,K2,CO1*
2. List the operations on languages. — *2,K1,CO1*
3. Define buffer pair. Why is buffering used in lexical analysis. — *2,K1,CO2*
4. Define lexemes. — *2,K1,CO2*
5. Give some examples of static checking? — *2,K1,CO4*
6. What is a type expression? — *2,K1,CO4*
7. Define DAG. — *2,K1,CO5*
8. What are the fields of activation record? — *2,K1,CO5*
9. List out the examples of function preserving transformations. — *2,K1,CO6*
10. What is an induction variable? — *2,K1,CO6*

PART - B (5 × 13 = 65 Marks)
Answer ALL Questions

11. a) Elaborate the phases of compiler and trace it with the program segment i=i*70+j+2. — *13,K2,CO1*

OR

b) (i) Demonstrate the language processing system with neat diagram. — *7,K2,CO1*
(ii) State the complier construction tools. Explain them. — *6,K2,CO1*

12. a) For the given Regular Expression construct DFA using Direct method (a|b)*abb. — *13,K3,CO2*

OR

b) Define Lex and Explain how lexical analyzer is constructed using lex? Give an example. — *13,K3,CO2*

13. a) Describe the various methods of implementing three-address statements. — *13,K2,CO4*

OR

K1 – Remember; K2 – Understand; K3 – Apply; K4 – Analyze; K5 – Evaluate; K6 – Create **12193**

b) How can Back patching be used to generate code for Boolean expressions and flow of control statements?   *13,K2,CO4*

14. a) Illustrate the various issues in the design of code generation in detail.   *13,K2,CO5*

OR

b) Explain the DAG representation of the basic block with an example.   *13,K2,CO5*

15. a) Explain briefly about the principal sources of optimization.   *13,K2,CO6*

OR

b) Explain global data flow analysis with necessary equations.   *13,K2,CO6*

PART - C (1 × 15 = 15 Marks)

16. a) Construct Stack implementation of shift reduce parsing for the grammar.   *15,K3,CO3*
E->E+E
E->E*E
E-> (E)
E->id and parse the input string id1+id2*id3.

OR

b) Construct LL(1) parsing table for the following grammar.   *15,K3,CO3*
S->iEtS | iEtSeS | a
E->b
Also check whether the given grammar is LL (1) or not?