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

Question Paper Code : 51353

B.E./B.Tech. DEGREE EXAMINATION, MAY/JUNE 2014.

Sixth Semester

Computer Science and Engineering

CS 2352/CS 62/10144 CS 602 — PRINCIPLES OF COMPILER DESIGN

(Regulation 2008/2010)

(Common to PTCS 2352 – Principles of Compiler Design for B.E. (Part-Time) Fifth Semester – Computer Science and Engineering – Regulation 2009)

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. State any two reasons as to why phases of compiler should be grouped.

2. Why is buffering used in lexical analysis? What are the commonly used buffering methods?

3. Define Lexeme.

4. Compare the features of DFA and NFA.

5. What is the significance of intermediate code?

6. Write the various three address code form of intermediate code.

7. Define symbol table.

8. Name the techniques in loop optimization.

9. What do you mean by Cross-Compiler?

10. How would you represent the dummy blocks with no statements indicated in global data flow analysis?

PART B — (5 × 16 = 80 marks)

11. (a) (i) Define the following terms : Compiler, Interpreter, Translator and differentiate between them. (6)

(ii) Differentiate between lexeme, token and pattern. (6)

(iii) What are the issues in lexical analysis? (4)

Or

(b) Explain in detail the process of compilation. Illustrate the output of each phase of compilation for the input "a = (b + c) * (b + c) * 2".

12. (a) Consider the following grammar

$S \rightarrow AS \mid b$

$A \rightarrow SA \mid a.$

Construct the SLR parse table for the grammar. Show the actions of the parser for the input string "abab".

Or

(b) (i) What is an ambiguous grammar? Is the following grammar ambiguous? Prove $E \rightarrow E + E \mid E * E \mid (E) \mid id$. The grammar should be moved to the next line, centered.

(ii) Draw NFA for the regular expression ab*/ab.

13. (a) How would you convert the following into intermediate code? Give a suitable example.

(i) Assignment statements. (8)

(ii) 'Case' statements. (8)

Or

(b) (i) Write notes on backpatching.

(ii) Explain the sequence of stack allocation processes for a function call.

14. (a) Discuss the various issues in code generation with examples.

Or

(b) Define a Directed Acyclic Graph. Construct a DAG and write the sequence of instructions for the expression a + a * (b – c) + (b – c) * d.

15. (a) Discuss in detail the process of optimization of basic blocks. Give an example.

Or

(b) What is data flow analysis? Explain data flow abstraction with examples.

————————