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

Question Paper Code : 70394

B.E./B.Tech. DEGREE EXAMINATIONS, NOVEMBER/DECEMBER 2021.

Sixth Semester

Computer Science and Engineering

CS 6660 — COMPILER DESIGN

(Common to Information Technology)

(Regulations 2013)

(Also common to : PTCS 6660 – Compiler Design for B.E. (Part-Time) – Computer Science and Engineering – Fifth Semester (Regulations – 2014))

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. Define the two parts of compilation.

2. List the cousins of the compiler.

3. List the rules that form the BASIS.

4. Differentiate tokens, patterns, lexeme.

5. What are the different error-recovery strategies that a parser can employ to recover from a syntactic error?

6. Define the function of left factoring.

7. How do you identify predictive parser and non-recursive predictive parser?

8. Name different storage allocation strategies used in run time environment.

9. What are the properties of optimizing compiler?

10. Write three address code sequence for the assignment Statement.
    $d := (a-b)+(a-c)+(a-c)$.

PART B — (5 × 13 = 65 marks)

11. (a) (i) Explain the phases of compiler with a neat diagram. (8)

(ii) Write notes on compiler Construction tools. (5)

Or

(b) (i) Explain the need for grouping of phases. (7)

(ii) Explain the various errors encountered in different phases of compiler. (6)

12. (a) (i) Explain the procedure for constructing a DFA from an NFA with example. (7)

(ii) Draw the transition graph for an NFA that recognizes the language aa*/bb*. (6)

Or

(b) (i) State and explain the procedure for constructing NFA from a regular expression. (7)

(ii) How to minimize the number of states of DFA? Explain it with an example. (6)

13. (a) Construct a predictive parsing table for the grammar

$S \rightarrow (L) \mid a$

$L \rightarrow L, S \mid S$

and show whether the following string will be accepted or not.

$(a,(a,(a, a)))$. (13)

Or

(b) Consider the following Grammar

$E \rightarrow E + T \mid T$

$T \rightarrow TF \mid F$

$F \rightarrow F^* \mid a \mid b$

construct the SLR parsing table for the above grammar. (13)

14. (a) (i) Describe about the contents of activation record. (6)

(ii) Create a parse trees for the following string : string id + id – id. Check whether the string is ambiguous or not. (7)

Or

(b) (i) Explain about various ways to pass a parameter in a function with example. (6)

(ii) Construct a Syntax-Directed Translation scheme that translates arithmetic expressions from infix into postfix notation. Using semantic attributes for each of the grammar symbols and semantic rules, Evaluate the input: 3*4+5*2. (7)

15. (a) Explain Principal sources of optimization with examples. (13)

Or

(b) (i) Explain various issues in the design of code generator. (7)

(ii) Write note on simple code generator. (6)

PART C — (1 × 15 = 15 marks)

16. (a) In SQL, keywords and identifiers are case-insensitive. Write a Lex program that recognizes the keywords SELECT, FROM, and WHERE (in any combination of capital and lower-case letters), and token ID, which may be any sequence of letters and digits, beginning with a letter. (15)

Or

(b) A simple matrix-multiplication program is given below:

```
for (i=0; i<n; i++)
  for (j=0; j<n; j++)
    c[i][j] = 0.0;
for (i=0; i<n; i++)
  for (j=0; j<n; j++)
    for (k=0; k<n; k++)
        c[i][j] = c[i][j] + a[i][k]*b[k][j] ;
```

(i) Translate the program into three-address statements. Assume the matrix entries are numbers that require 8 bytes, and that matrices are stored in row-major order. (7)

(ii) Construct the flow graph for the code from 1. (6)

(iii) Identify the loops in the flow graph from 2. (2)