Reg. No. : □□□□□□□□□□□□

Question Paper Code : X 20407

B.E./B.Tech. DEGREE EXAMINATIONS, NOVEMBER/DECEMBER 2020
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 –
Fifth Semester – Computer Science and Engineering – Regulations 2014)

Time : Three Hours Maximum : 100 Marks

Answer ALL questions

PART – A **(10×2=20 Marks)**

1. What is the usage of sentinel in lexical analyzer ? List its advantages.

2. Construct regular expression for the binary string that starts with 0 and has odd length or that starts with 1 and has even length.

3. Consider the following grammar and demonstrate that the grammar is ambiguous by showing two different parse trees for some string.

   $S \rightarrow AB \mid aaB$

   $A \rightarrow a \mid Aa$

   $B \rightarrow b$

4. Consider the following grammar :

   $S \rightarrow A$

   $A \rightarrow A + A \mid B{+}{+}$

   $B \rightarrow y$

   Show a leftmost and rightmost derivation for the string “y + + + y + +”.

5. Write the semantic action for the production rule of E ->E1 OR M E2.

6. Translate the arithmetic expression x = (a + b)*– c/d into quadruples and triples.

7. How to perform register assignment for outer loops ?

8. Write the static single assignment form for the below code segment and write the minimum number of total variables required for the conversion.

   x = u – t;

   y = x * v;

   x = y + w;

   y = t – z;

   y = x * y;

9. Define static allocations and stack allocations.

10. Why are quadruples preferred over triples in an optimizing compiler ?

PART – B **(5×13=65 Marks)**

11. a) i) What are the different error recovery strategies in phases of a compiler ? **(5)**

   ii) What are the tools used for constructing a compiler ? **(8)**

   (OR)

   b) What do you mean by passes and phases of a compiler ? Explain the various phases of a compilation with neat diagram. **(13)**

12. a) Draw an NFA and then an equivalent DFA for the regular expression : (10 | 01) * | 100 and also construct optimized DFA using the subset construction followed by Hopcroft's minimization procedure. **(13)**

   (OR)

   b) i) Discuss the issues involved in designing lexical analyser. **(8)**

   ii) Describe in detail about input buffering. **(5)**

13. a) i) Explain dangling-else problem with an example and give the unambiguous grammar for the same. **(7)**

   ii) Consider the following grammar. Parse the input string "abbcde" using stack implementation of shift-reduce parser. **(6)**

   $S \rightarrow aABe$

   $A \rightarrow Abc \mid b$

   $B \rightarrow d$

   (OR)

b) Construct LL (1) parsing table for the following grammar using FIRST and FOLLOW set. **(9)**

S → UVW

U→ (S) | aSb | d

V → aV | e

W → cW | e

Give the parsing actions for the input string "(dc)ac". **(4)**

14. a) Below is a grammar for expressions involving operator + and integer or floating-point operands. Floating-point numbers are distinguished by having a decimal point.

E -> E + T | T

T -> num.num | num

i) Give an Syntax Directed Definition (SDD) to determine the type of each term T and expression E. **(4)**

ii) Extend your SDD of (i) to translate expressions into postfix notation. Use the binary operator intToFloat to turn an integer into an equivalent float. **(4)**

iii) Give an SDD to differentiate expressions such as x*(3*x + x*x) involving the operators + and *, the variable x, and constants. Assume that no simplification occurs, so that, for example, 3 * x will be translated into 3 * 1 + 0 * x. Note : differentiation (x * y) = (x * differentiation(y) + differentiation(x) * y) and differentiation(x + y) = differentiation(x) + differentiation(y). **(5)**

(OR)

b) Write the syntax directed translation scheme with backpatching to generate three address code for the given grammar. **(13)**

S → while E do S | begin L end

L → L; S | S

E → E or E | E and E | not E | id

15. a) Discuss in detail the role of dead code elimination and loop optimization during code optimization of a compiler. **(13)**

(OR)

b) Explain the issues in code generation phase of a compiler. **(13)**

PART – C **(1×15=15 Marks)**

16. a) Construct CLR (1) parsing table for the following grammar. **(15)**

And prove that it is not SLR (1).

$S \rightarrow Aa$

$S \rightarrow dAb$

$S \rightarrow dca$

$S \rightarrow cb$

$A \rightarrow c$

(OR)

b) i) Construct a Syntax-Directed Translation scheme that translates arithmetic expressions from infix into postfix notation. Your solution should include the context-free grammar, the semantic attributes for each of the grammar symbols and semantic rules. Show the application of your scheme to the input string "3 ∗ 4 + 5 ∗ 2". **(10)**

ii) Use the following code to identify the leader instructions and their corresponding basic blocks and draw the control flow graph below. **(5)**

```
 1)  P : = 0
 2)  I : = 1
 3)  P : = P + I
 4)  IF P < = 60 GOTO (7)
 5)  P : = 0
 6)  I : = 5
 7)  T1 : = I ∗ 2
 8)  I : = T1 + 1
 9)  IF I <= 20 GOTO (3)
10)  K : = P ∗ 3
```