O–51961

Register No. | | | | | | | | |

BE Degree Examination April 2019

Sixth Semester

Computer Science and Engineering

14CST61 – COMPILER DESIGN

(Regulations 2014)

Common to BTech Information Technology

Time: Three hours — Maximum: 100 marks

Answer all Questions

Part – A (10 × 2 = 20 marks)

1. List the compiler construction tools. [CO1,K1]
2. Compare Token, Pattern and Lexeme. [CO1,K2]
3. Consider the context free grammar $S \rightarrow SS+ \mid SS^* \mid a$ and the string aa+a*. Is the grammar ambiguous or unambiguous? Justify. [CO1,K3]
4. Define handle pruning with an example. [CO1,K2]
5. Translate the arithmetic expression a+a*(b-c)+(b-c)*d to postfix notation and draw DAG. [CO3,K2]
6. Suppose that a is an array of integers and that f is a function from integers to integers. Then the assignment is $n = f(a[i])$. Generate the three address code for this statement. [CO3,K3]
7. Determine the liveness and next use information for the statement i:x=y+z. [CO4,K2]
8. Compute the costs of the following instruction sequence [CO4,K3]
   LD R0, i
   MUL R0, R0, 8
   LD R1, a(R0)
   ST b,R1
9. State the use of registers and address descriptors in code generation. [CO5,K1]
10. Generate the assembly code for the following 'C' statement [CO5,K3]
    x=a-d*e | f

Part – B (5 × 13 = 65 marks)

11. a. i) Outline the various phases of compiler and show how the following expression is translated to target code in each phase (8) [CO1,K2]
$x = y + 2.0 * z$.

ii) Draw the $\in$–NFA for the given regular expression using Thompson's construction method (5) [CO1,K3]
(a/b)* a (a | b) (a | b).

(OR)

b. i) Write a lex program that identifies the tokens in the given statement (5) [CO2,K3]
$a = b + c * d - 52.3$

ii) Construct a minimised DFA for the regular expression $(a \mid \in)^* a(a \mid b)$. (8) [CO1,K3]

12. a. Construct the predictive parsing table for the grammar (13) [CO1,K3]
$S \rightarrow S(S)S \mid \in$ and parse the behaviour for the string (( )).

(OR)

b. Check whether the given grammar is LR(0) (13) [CO4,K3]
$E \rightarrow E + T / T \quad T \rightarrow id/(E)$. If it is LR(0), show the behaviour of an example string in the parser.

13. a. i) Write the intermediate code for the statement $a = (x - y) * z/(x - y)$ and implement it using the three methods. (7) [CO3,K2]

ii) Write the syntax directed translation scheme and make use of it to generate the three address statement for $a = b + -c$. (6) [CO3,K3]

(OR)

b. Generate the intermediate code for the following code segment along with the required syntax directed translation scheme (13) [CO3,K3]

```
while (i <= 10)
{
    a[i]=0;
    i=i+1;
}
```

14. a. i) Paraphrase on the issues in design of code generation. (6) [CO4,K1]

ii) Dwell briefly on peephole optimization with examples. (7) [CO4,K2]

(OR)

b. Elaborate on the principal sources of optimization with suitable examples. (13) [CO4,K2]

15. a. Discuss in detail about dynamic storage allocation of compilers using stack and heap storage. (13) [CO5,K1]

(OR)

b. i) Summarise the actions and basic operations performed by optimizing compilers for garbage collector to find the correct root set and to change the set of reachable objects. (7) [CO5,K1]

ii) Illustrate the code generation for simplified procedure call and returns using static allocation. (6) [CO5,K2]

Part – C (1 × 15 = 15 marks)

16. a. Construct a LR(1) parser for the following grammar and show that it is LR(1) but not LALR(1) (15) [CO1,K3]
$S \rightarrow Aa \mid bAa \mid Bc \mid bBa$
$A \rightarrow d$
$B \rightarrow d$

(OR)

b. i) Apply backpatching and generate the intermediate code for (8) [CO3,K3]
x<y && x>100' | | 'x>y
Write the semantic rules and derive the parse tree for the above statement.

ii) Identify the loops and basic blocks for the following statements (7) [CO4,K3]
for (i = 1; i<=n; i++)
sum = sum+a[i]

Bloom's Taxonomy Level	Remembering (K1)	Understanding (K2)	Applying (K3)	Analysing (K4)	Evaluating (K5)	Creating (K6)
Percentage	16.67	27.22	56.11	-	-	-