CS61

USN	1	M	S							

RAMAIAH Institute of Technology

(Autonomous Institute, Affiliated to VTU)
(Approved by AICTE, New Delhi & Govt. of Karnataka)
Accredited by NBA & NAAC with 'A+' Grade

SEMESTER END EXAMINATIONS – MAY 2023

Program	:	**B.E. – Computer Science and Engineering**	**Semester**	:	**VI**
Course Name	:	**Compiler Design**	**Max. Marks**	:	**100**
Course Code	:	**CS61**	**Duration**	:	**3 Hrs**

Instructions to the Candidates:

- Answer one full question from each unit.

UNIT - I

1.	a)	Define the terms lexeme, pattern and token with examples. Identify the lexemes that make up the tokens in the following C program int max (int i, int j) /* return maximum of i and j */ {return i>j? i:j;}	CO1	(06)
	b)	Discuss how Input buffering is handled by lexical analyzer using sentinels and buffer pairs.	CO1	(07)
	c)	Construct the transition diagrams for: i) Hide, side, tide, said ii) Logical operators.	CO1	(07)
2.	a)	Explain the various phases of compiler. Give the translation of the assignment statement: a=b+c*10 at the end of each phase. Assume a,b,c as float data types.	CO1	(09)
	b)	What is a left-recursion7 Write the algorithm to eliminate left recursion and apply it to eliminate left-recursion from the following grammar A→Ba | Aa | c B→Bb |Ab |d	CO1	(06)
	c)	Illustrate the different lexical errors and recovery actions.	CO1	(05)

UNIT - II

3.	a)	Describe the shift reduce parsing of SR parser for arithmetic expression for the input string : (id+id)*id.	CO2	(08)
	b)	Construct the CLR parser's canonical collection of items for the following grammar. S-->AA A-->aA|b	CO2	(07)
	c)	Mention the steps for LALR parser construction from CLR parser.	CO2	(05)
4.	a)	Illustrate the algorithms of FIRST and FOLLOW for Predictive parser using the following grammar. S→A | a A→a	CO2	(08)
	b)	Compare SLR, CLR and LALR parsers.	CO2	(05)
	c)	Apply SLR parser construction algorithms to construct SLR parser for the grammar: S→ (S)S | ϵ.	CO2	(07)

UNIT - III

5.	a)	What is meant by calling sequence and return sequence? List calling sequence design principles.	CO3	(08)
	b)	Define a syntax directed definition. Give SDD for the simple type declaration including int and float types.	CO3	(06)
	c)	Illustrate how infix to prefix translation is implemented with the help of semantic actions.	CO3	(06)

6.	a)	Define synthesized and inherited attributes. Give example for each.	CO3	(04)
	b)	List the different storage allocation strategies and explain in detail.	CO3	(10)
	c)	Define postfix SDT's. Write the rules for turning an L- attributed SDD into an SDT.	CO3	(06)

UNIT- IV

7.	a)	Write about the value number method for constructing DAG's for an expression with example.	CO4	(08)
	b)	Analyze the method of computing data types and their width along with declaration grammar.	CO4	(07)
	c)	Define the semantic rules to generate intermediate code for flow of control statements.	CO4	(05)
8.	a)	Specify the instruction forms of three address codes and give examples for each one of them.	CO4	(08)
	b)	Explain the translation of arithmetic expressions using semantic rules.	CO4	(07)
	c)	Appraise about the translation of a break, continue and goto statements to intermediate codes.	CO4	(05)

UNIT - V

9.	a)	Briefly explain the main issues in code generation.	CO5	(10)
	b)	Describe the following terms with an example. i) Common sub expression elimination ii) Dead code elimination.	CO5	(10)
10.	a)	What is basic block and flow graph and Generate three address code, basic block and flow graph for the following C segment.	CO5	(10)

```
for  (i=1; i<=10; i++)
for ( j=1; j<=10; j++)
    a[i][j]=0;
for (i=1; i<=10; i++)
    a[i][i]=1;
```

	b)	Explain the code generation algorithm and generate code for the following expression. X = a/(b + c) – d* (e + f).	CO5	(10)

****