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 – JULY / AUGUST 2022

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)	Examine the working principles of lexical analyzer and semantic analyzer with appropriate examples.	CO1	(08)
	b)	Relate the compiler and interpreter in context of program translations for machine code executions.	CO1	(05)
	c)	Appraise about the phases of compiler along with analysis and synthesis part break ups.	CO1	(07)
2.	a)	Paraphrase about prefix, suffix, and substring subsequences for the string "Ramaiah Institute of Technology".	CO1	(08)
	b)	Demonstrate the elimination of left recursion in an ambiguous grammar with appropriate example.	CO1	(07)
	c)	Illustrate the pattern usage for tokens recognition and examine the input string match with the patterns.	CO1	(05)

UNIT – II

3.	a)	Consider the following Context Free Grammar **S→SA |0 |ε** **A→aS1 | a** i. Check whether the Grammar is suitable for non-recursive predictive parser. If not, make it suitable and Compute the FIRST and FOLLOW sets for each non-terminal symbol. ii. Construct the parsing Table for a non-recursive predictive parser for the grammar. iii. Is the Grammar LL (1)? Justify.	CO2	(09)
	b)	Write an algorithm to construct CLR (1) parse table.	CO2	(05)
	c)	Demonstrate that the following grammar is not SLR. Justify why the grammar can't be in LR (0). **E→1E1 | 1**	CO2	(06)
4.	a)	Write down the rules to check whether a Grammar is LL (1) or not. Give suitable example of a LL (1) grammar.	CO2	(05)
	b)	Prove that the given Context Free Grammar is LR (1) by generating the parse table. **S→AaAb | BbBa** **A→ε** **B→ε** Show the actions made by the parser on validating the input string **'ab$'**.	CO2	(10)
	c)	Explain the need of Augmentation in LR grammars and how the given grammar can be changed to an augmented grammar?	CO2	(05)

UNIT – III

5.	a)	Classify the two kinds of attributes for nonterminal in a grammar. Give examples and show them.	CO3	(08)
	b)	Appraise about the Construction of syntax trees for simple expressions.	CO3	(07)
	c)	Classify the Typical run-time memory into code and data areas. Mention their associate uses.	CO3	(05)
6.	a)	Show that the Dependency graphs are useful tool for determining the evaluation order for the attribute instances in a parse tree.	CO3	(08)
	b)	Show the Parser stack with a field for synthesized attributes using a bottom-up parsing stack along with an example.	CO3	(07)
	c)	Illustrate the activation tree and activation record involvements for functions calls.	CO3	(05)

UNIT – IV

7.	a)	Describe the process of generating three address code for flow control statements with help of Annotated Parse tree for **if(a<0 && a>5 || !a) a=1;**	CO4	(07)
	b)	Illustrate with an example how the computation of type and relative address for struct datatype is done. Consider the grammar given below **P→D** **D→T id ; D_1 | ε** **T→ struct { D } | int**	CO4	(07)
	c)	Draw DAG, obtain three address code and specify the Value Number method representation for the expression given below **a=a+a + a + b + (a+b) +(a+b)**	CO4	(06)
8.	a)	Write the three address code and quadruple representation for the given statements (Assume all variables are integer) i. f=min(1,n-1,n+1) ii. a=x[i]+b*c	CO4	(08)
	b)	Illustrate the Translation Scheme using Backpatching for control flow statements. Discuss the three functions are used for manipulating list of jumps in the process of translation.	CO4	(06)
	c)	Discuss about the computation of type and width of array datatype for the input **int a[3]** by drawing an annotated parse tree. Is the grammar given an attribute grammar? If not state the reason. **G: D→ T id L** **T→ int | float** **L→L_1, id | id**	CO4	(06)

UNIT – V

9.	a)	Label the factors for code generator which are useful to map the intermediate representation of program. Write about them.	CO5	(08)
	b)	State about the code generation for simplified procedure calls and returns using three address statements.	CO5	(07)
	c)	Demonstrate with examples the construction of directed acyclic graph for basic blocks.	CO5	(05)
10.	a)	Classify the kinds of instructions available for simple target machine model. Brief about them.	CO5	(08)
	b)	Write an algorithm to partition three address instructions into basic block blocks. Demonstrate it with an example.	CO5	(07)
	c)	Discuss the four principle uses of registers in a simple code generator.	CO5	(05)

****