U.S.N.										

B.M.S. College of Engineering, Bengaluru-560019

Autonomous Institute Affiliated to VTU

September / October 2024 Supplementary Examinations

Programme: B.E. **Semester: V**
Branch: Computer Science and Engineering **Duration: 3 hrs.**
Course Code: 22CS5PCCPD **Max Marks: 100**
Course: Compiler Design

Instructions: 1.Answer any FIVE full questions, choosing one full question from each unit.
2. Missing data, if any, may be suitably assumed.

Important Note: Completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages. Revealing of identification, appeal to evaluator will be treated as malpractice.

		UNIT - I	***CO***	***PO***	**Marks**
1	a)	Explain the different phases of compiler and analyze the given line of code and show how this line gets converted to target code in assembly language. position = initial + rate * 60.00	*CO2*	*PO2*	**10**
	b)	Outline the benefits of using sentinels? Give Look-ahead code for sentinels.	*CO2*	*PO2*	**6**
	c)	Construct a transition diagram for identifying: (i) identifiers (ii) whitespaces	*CO1*	*PO1*	**4**
		UNIT - II			
2	a)	Eliminate Left recursion from the grammar given below: bexpr -> bexpr or bterm | bterm bterm -> bterm and bfactor | bfactor bfactor -> not bfactor | (bexpr) | true | false	*CO1*	*PO1*	**5**
	b)	Analyze and design a predictive parser for the grammar E -> 5 + T | 3 - T T -> V | V * V | V + V V -> a | b Also construct the Predictive parsing table	*CO2*	*PO2*	**5**
	c)	Analyze and design a LR(0) parser for the grammar S -> S(S)S S -> € And also parse the string " (( ))" give state transition diagram as well.	*CO3*	*PO3*	**10**
		OR			
3	a)	Design LR(0) parser for the grammar S -> SS+ | SS* | a And parse the string "aa+a*" give state transition diagram as well.	*CO3*	*PO3*	**10**

		b)	Design a LR(1) parser and LALR parser for the grammar. Represent a finite state machine diagram. S -> CC C -> cC | d And parse the string "ccd" for CLR and LALR parser.	*CO3*	*PO3*	**10**
			UNIT - III			
	4	a)	Give the SDD for Simple Desk Calculator and draw the dependency graph for expression 1*2*3*(4+5)n Also outline the type of attribute in each production.	*CO1*	*PO1*	**10**
		b)	Give SDT for simple arithmetic expression using top down approach(L-attributed definition) for a-4+c.	*CO1*	*PO1*	**10**
			UNIT-IV			
	5	a)	Explain the various 3 address instructions form. Give examples for each type of instruction.	*CO2*	*PO2*	**7**
		b)	Give the 3 address code for the following: (i) -(a*b)+(c+d)-(a+b+c+d) (ii) if a< b and c< d then t=1 else t=0	*CO3*	*PO3*	**7**
		c)	Define Quadruples, Triples and Indirect Triples. Translate a+-(b+c) into Quadruples, Triples and Indirect Triples form.	*CO1*	*PO1*	**6**
			OR			
	6	a)	Give SDT for Boolean Expressions.	*CO3*	*PO3*	**6**
		b)	Write the SDD for the following flow control statements: (i) if (ii) if-else (iii) while	*CO2*	*PO2*	**6**
		c)	Obtain DAG for a + a * (b-c) + (b-c)*d. Also give the steps of construction.	*CO3*	*CO3*	**8**
			UNIT - V			
	7	a)	Generate three address code, basic block and flow graph for the following code snippet: sum=0 for(i=0;i<=10;i++) sum=sum + a[i]	*CO3*	*PO3*	**10**
		b)	With a simple procedure call example explain static and stack allocation.	*CO2*	*PO2*	**10**

****