Set 1

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

USN

School of Computer Science and Engineering

B.Tech (Hons.)

Midterm Question Paper

Academic Year 2024-2025

Course: Compiler Design		Course Code: CS3704	Semester: VI
Time:	Duration: 90 minutes	Date :	Max Marks: 25

Answer all questions

Sl. No.	PART A – (MCQs) Max Marks(5)	Marks	L1-L6	CO
1.	A grammar with production rules { A→ Ba | Cb, B□CA, C□ c | ε} contains A. Left factor **B. Left recursion** C. Both left factor and left recursion D. None of the other options	1	L3	CO2
2.	Derivation produced by a top-down parser is **A. Leftmost** B. Rightmost C. Either leftmost and rightmost D. None of the other options	1	L2	CO2
3.	A grammar is ambiguous if A. Its left most and right most derivations are different **B. More than one left most derivations exist** C. There is no left most derivation D. There is no rightmost derivation	1	L2	CO1
4.	Which of the following has more expressive power? A. DFA B. NFA C. Regular expression **D. All the mentioned**	1	L2	CO1
5.	Between NFA and DFA which is powerful in recognizing language A. NFA B. DFA **C. Equally powerful**	1	L2	CO1

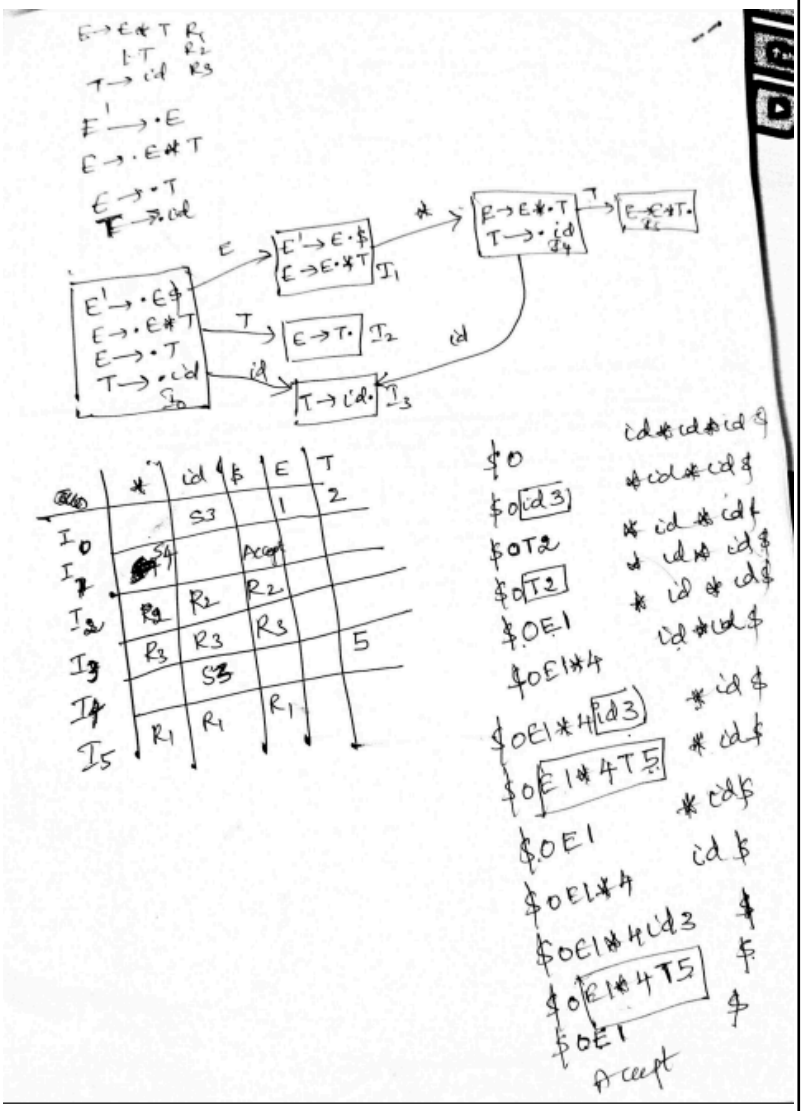
L1- Remember, L2- Understand, L3 – Apply, L4- Analyse, L5 – Evaluate, L6 – Create

	D. Cannot be said definitely			

Sl. No.	PART B – Max Marks (20)	Marks	L1-L6	CO

6.	A developer writes a program in C, but after compilation, they notice a syntax error, followed by a type mismatch warning. Later, after fixing these issues, the program runs but produces incorrect output due to an optimization-related issue. Explain which phases of the compiler are responsible for detecting these errors and optimizing the code.
	
	Justify your answer with sample code.
	
	Syntax Error – Handled by the Syntax Analysis Phase
	
	int main() { printf("Hello, world!" // Missing closing parenthesis }
	
	Type Mismatch Warning – Handled by the Semantic Analysis Phase
	
	int x = "hello"; // Type mismatch: assigning string to an integer
	
	The **code optimization phase** improves performance by transforming the intermediate code, but incorrect optimizations may alter program behavior.
	
	#include <stdio.h>
	
	int main() {
	
	int arr[5] = {1, 2, 3, 4, 5};
	
	int sum = 0;
	
	for (int i = 0; i <= 5; i++) { // Off-by-one error (out-of-bounds access)
	
	sum += arr[i];
	
	}
	
	printf("Sum: %d\n", sum);
	
	return 0;
	
	}
	
	Explanation +justification sample code for each phase ---(1.5 each * 3)+ listing the phases(0.5 marks)
		5	L4	CO1

7.	Consider the following context-free grammar where the start symbol is S and the set of terminals is $\{a,b,c,d\}$. $S \rightarrow AaAb \mid BbBa$ $A \rightarrow cS \mid \epsilon$ $B \rightarrow dS \mid \epsilon$ Check whether the grammar is LL(1) or not by constructing parsing table. First ---1.5 Follow 1.5 Table-----1.5 mark (each row----0.5 marks) It is LL(1) grammar-----0.5 marks To complete the given LL(1) table first we have to find the FIRST and FOLLOW of the given grammar, that is: | | FIRST | FOLLOW | | $S \rightarrow AaAb \mid BbBa$ | $\{a,b,c,d\}$ | $\{\$,a,b\}$ | | $A \rightarrow cS \mid \varepsilon$ | $\{c,\varepsilon\}$ | $\{a,b\}$ | | $B \rightarrow dS \mid \varepsilon$ | $\{d,\varepsilon\}$ | $\{a,b\}$ | Now we can fill the entries in LL(1) table: | | a | b | c | d | \$ | | S | $S \rightarrow AaAb$ | $S \rightarrow BbBa$ | $S \rightarrow AaAb$ [1] | $S \rightarrow BbBa$ [2] | | | A | $A \rightarrow \varepsilon$ | $A \rightarrow \varepsilon$ [3] | $A \rightarrow cS$ | | | | B | $B \rightarrow \varepsilon$ [4] | $B \rightarrow \varepsilon$ | | $B \rightarrow dS$ | |	**5**	**L3**	**CO2**
8.	Consider the following grammar G, with S as the start symbol. Compute the FIRST and FOLLOW sets for the non-terminals in the grammar S□ daT | Rf T□aS | bS | ϵ R □cTR | ϵ First---2.5 marks follow 2.5 marks | | FIRST | FOLLOW | | $S \rightarrow daT \mid Rf$ | $\{c,d,f\}$ | $\{\$,c,f\}$ | | $T \rightarrow aS \mid bS \mid \varepsilon$ | $\{a,b,\varepsilon\}$ | $\{\$,c,f\}$ | | $R \rightarrow cTR \mid \varepsilon$ | $\{c,\varepsilon\}$ | $\{f\}$ |	**5**	**L3**	**CO2**
9.	Check whether the grammar is LR(0) or not by parsing input id * id *id	**5**		

	E→E*T | T T→id  Transition diagram—1.5 Table-1.5 Parsing---1.5 It is lr(0) justifying 0.5 marks		**L3**	**CO2**

Course Outcomes

1. Develop skills to devise, select, and apply appropriate tools and techniques for effective compiler design.
2. Apply context-free grammars (CFG) to develop language specifications.
3. Analyze syntax-directed translation schemes for various programming constructs and generate intermediate code.
4. Develop knowledge about run-time data structures like symbol table organization and different techniques.
5. Apply advanced knowledge of compiler optimization and code generation to practical scenarios.

Marks Distribution									
L1	L2	L3	L4	L5	L6	CO1	CO2	CO3	CO4
	4	16	5			8	17		

L1- Remember, L2- Understand, L3 – Apply, L4- Analyse, L5 – Evaluate, L6 – Create