BIRLA INSTITUTE OF TECHNOLOGY, MESRA, RANCHI
(END SEMESTER EXAMINATION)

CLASS:	**MCA**	**SEMESTER : III**
BRANCH:	**MCA**	**SESSION : MO/2023**

SUBJECT: CA513 COMPILER DESIGN

TIME:	**3 Hours**	**FULL MARKS: 50**

INSTRUCTIONS:
1. The question paper contains 5 questions each of 10 marks and total 50 marks.
2. Attempt all questions.
3. The missing data, if any, may be assumed suitably.
***4*. Before attempting the question paper, be sure that you have got the correct question paper.**
5. Tables/Data hand book/Graph paper etc. to be supplied to the candidates in the examination hall.

			CO	**BL**
Q.1(a)	Write some important features of any standard Compiler.	[5]	CO-1	Remember
Q.1(b)	For C-like statement - int a, *if*, b; Will we get any lexical error from this statement? Justify your answer. If not, then at which phase, error will be detected? Explain in details.	[5]	CO-1	Apply
Q.2(a)	Is x + y; a valid C-statement, where *x*, *y* are the identifiers? If it is valid, write down the CFG for recognizing such statement.	[5]	CO-4	Apply
Q.2(b)	Explain the *issues* in Top-down parsing. Find LL(1) table for the grammar G: S→aAb , A→cd |c, where *a, b, c* and *d* are the tokens, and S and A are the non-terminals with S as the start symbol. Check whether the grammar is LL(1) or not.	[5]	CO-4	Understand
Q.3(a)	Consider the expression grammar(G): E→E+E , E→id, where E is the non-terminal symbol and *id* is the token. Find CLR(1) Table for this grammar.	[5]	CO-2	Apply
Q.3(b)	Explain how LALR(1) parser resolves R/R and S/R conflicts.	[5]	CO-3	Understand
Q.4(a)	Design CFG for *declaration statements* of C-language, assuming data types - *integer*, *floating-point* and *character* (you need not consider *size* and *sign* specifiers).	[5]	CO-4	Understand

Q.4(b)	Design SDT for the CFG designed for Q.-no. 4(a).
	Explain the importance of Intermediate code. Construct Flow Graph for the following C-code segment.
	int i;
	for(i=0; ++i; i++)
	printf("%d", i);
		[5]	CO-4	Understand

Q.5(a)	Discuss how memory is managed at the time of execution of a program.	[5]	CO-2	Understand
Q.5(b)	Explain briefly how code optimizer optimizes code.	[5]	CO-2	Understand

:::::22/11/2023 E:::::