Code No.: CS601PC	R20	H.T.No.			8	R						

CMR ENGINEERING COLLEGE: : HYDERABAD
UGC AUTONOMOUS
III–B.TECH–II–Semester End Examinations (Supply) - December- 2025
COMPILER DESIGN
(Common for CSE, IT, CSC, CSD)

[Time: 3 Hours] **[Max. Marks: 70]**

Note: This question paper contains two parts A and B.
Part A is compulsory which carries 20 marks. Answer all questions in Part A.
Part B consists of 5 Units. Answer any one full question from each unit. Each question carries 10 marks and may have a, b, c as sub questions.

PART-A **(20 Marks)**

1. a)	List out the six phases of the compiler.	[2M]
b)	Differentiate between Lexeme and token.	[2M]
c)	Explain conflicts in shift reduce parsing.	[2M]
d)	Define goto function in LR parser with an example?	[2M]
e)	Define syntax tree? Draw the syntax tree for the assignment statement. a :=b * -c + b * -c.	[2M]
f)	Show the DAG for a: =b *-c + b * -c?	[2M]
g)	What do you mean by machine dependent and machine independent optimization?	[2M]
h)	Explain the role of code generator in a compiler?	[2M]
i)	Define the term copy propagation?	[2M]
j)	Mention the issues to be considered while applying the techniques for code Optimization?	[2M]

PART-B **(50 Marks)**

2.	For the following expression Position:=initial+ rate*60 Write down the output after each phase of compiler.	[10M]
	OR	
3.	Construct Deterministic Finite Automata to accept the regular expression : (0+1)* (00+11) (0+1)*	[10M]
4.	Consider the grammar E -> E + E|E *E|(E)| id Show the sequence of moves made by the shift-reduce parser on the input id1+id2*id3 and determine whether the given string is accepted by the parser or not.	[10M]
	OR	
5.	Prepare a canonical parsing table for the grammar given below S-> CC C->cC|d.	[10M]
6.a)	Write a note on the specification of a simple type checker.	[5M]
b)	Define a type expression? Explain the equivalence of type expressions with an appropriate example.	[5M]
	OR	
7.	Write quadruples, triples and indirect triples for the expression: -(a*b)+(c+d)-(a+b+c+d)	[10M]
8.	Write a short note on: a. Flow graph (with example) b. Dominators c. Natural loops d. Inner loops e. Reducible flow graphs.	[10M]
	OR	
9.	Demonstrate code generation algorithm in detail.	[10M]
10.	Explain Local optimization and loop optimization in detail.	[10M]
	OR	
11.	Explain how loop invariant computation can be eliminated? Explain how "Redundant sub-expression eliminates" can be done in a given program?	[10M]

****