Reg. No. : | | | | | | | | | | | | |

Question Paper Code: R5D02

B.E./B.Tech. DEGREE EXAMINATION, NOV 2025

Fifth Semester

R21UCB502 COMPILER DESIGN

(Regulations R2021)

Duration: Three hours Maximum: 100 Marks

Answer ALL Questions

PART A - (10 x 2 = 20 Marks)

1.	Define patterns and lexemes with suitable example.	CO1- U
2.	(ab/ab)* Convert regular expression in to finite automata using Thompsons Construction.	CO 6- App
3.	Give short note on Grammar Types.	CO 2- App
4.	Define RDP.	CO 2- Ana
5.	Generate the intermediate representation of the statement . A or B and not C.	CO 6- Ana
6.	How semantic Errors can be detected?	CO 6- U
7.	Define Basic Block and Flow Graph.	CO 6- U
8.	Give the application of DAGs.	CO 6- U
9.	Write the actions to perform the code generation algorithms.	CO 2- U
10.	Mention any two objectives of a good code generator.	CO 2- U

PART – B (5 x 16= 80 Marks)

11.	(a)	(i) Explain the Role of Lexical Analyzer.	CO1-U	(8)
		(ii) Write notes on specification of tokens.		(8)
		Or		
	(b)	Explain various phases of compiler with neat sketch.	CO1-U	(16)

12. (a) Consider the following grammar CO2- App (16)

E →E + T | T
T → TF | F
F → F* | a | b

Construct the SLR parsing table for this grammar and also parse the input a * b + a

Or

(b) Check whether the following grammar is SLR (1) or not. Explain your answer with reasons. CO2- App (16)

E→E+E
E-> (E)
E→ E*E
E→id

13. (a) i) Draw the quadruple, triple and indirect triples for the following statement a = b * – c + b * – c. CO1-U (16)

(ii) Draw the quadruple, triple and indirect triples for the following statement (a + b) * (c + d) - (a + b + c).

Or

(b) Translate the statement CO2-U (16)

```
c=0
do{
    if (a<b) then
            x++
    else
            x - -
            c + +
} while(c<5)
```

into three address statements.

ii)Convert the arithmetic expression ((A + B) * C) - (D / E) into postfix and prefix notations. Then, translate each notation into three-address code. Discuss the utility of postfix and prefix notations in compiler design.

14. (a) Describe the various types of peep-hole optimization techniques with examples CO1-U (16)

Or

(b) Explain the role of basic blocks and control flow graphs in data flow analysis. CO1-U (16)

15. (a) Explain the role of stack allocation in managing procedure calls and local variables. CO1-U (16)

Or

(b) Explain the basic structure and components of a simple code generator. CO1- U (16)