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

Question Paper Code : 40454

B.E./B.Tech. DEGREE EXAMINATIONS, NOVEMBER/DECEMBER 2025.

Sixth Semester

Computer Science and Engineering

CS 8602 — COMPILER DESIGN

(Common to Computer Science and Business Systems)

(Regulations 2017)

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. Compare and contrast Interpreter and Compiler.

2. Define the role of input buffering in lexical analysis.

3. Design a CFG for the language over $\Sigma = \{a, b\}$ such that :
   - The string starts and ends with the same symbol,
   - The total number of symbols is at least 5,
   - And no two consecutive b's are allowed in the string.

4. List out the steps in performing LL parsing.

5. What do you mean by Syntax directed translation? Write SDD for declarative statements.

6. What is the significance of Three address code?

7. When do you call a variable to be syntactically live at a point?

8. When does dangling references mean?

9. List out the peephole optimization technique.

10. What do you understand by global data flow analysis?

PART B — (5 × 13 = 65 marks)

11. (a) Every high-level programming statement needs to be converted into machine code before execution. With a neat sketch, explain the translation process involved. Use the following Python statement as an example :

if(x == y or z != 0): print("Access Granted")

Or

(b) The employees of a company access the building through a two-step verification system :

Step 1 : Swipe their employee ID card.

Step 2 : Enter a 7-character security code on the keypad following these rules :

(i) The first three characters represent the employee department code (three uppercase letters).

(ii) The fourth character is either 'S' or 'J' indicating Senior or Junior level.

(iii) The fifth and sixth characters represent the current month in two digits (01 to 12).

(iv) The seventh character is 'E' for entry or 'L' for leaving the building.

Design a regular expression that represents this security code format. Construct a DFA that recognizes whether the input code is valid according to the pattern. Validate your DFA with two example strings.

12. (a) Consider the following Context Free Grammar, G :

$S \rightarrow 0A1 \mid a$

$A \rightarrow A*S \mid B$

$B \rightarrow a$

Construct LL(1) parsing table. Check whether the following word, "0a*a1" is accepted or not. Show the stack status, input and rules used for parsing the input string.

Or

(b) Consider the following grammar :

$S \rightarrow A \mid B$

$A \rightarrow aAb \mid C$

$B \rightarrow bBb \mid C$

$C \rightarrow pqrW$

$W \rightarrow d$

(i) Construct SLR parsing table for G. (10)

(ii) Show the stack status, input and shift/reduce action used for parsing the string "apqrdb". (3)

13. (a) Convert the following statement into three address statements. Represent the converted three address codes in various possible ways.

Grammar G

$S \rightarrow E\%F \mid F$

$F \rightarrow F@T \mid T$

$T \rightarrow D$

$D \rightarrow 0|1|2|3|4|5|6|7$

Word, W :

5%6%3@4@3

Or

(b) Write the syntax directed translation for the following piece of code.

white a < b or c > d

do

if c > d

then

x : = y + z

else

x : = y – z

14. (a) Elaborate the issues in the design of a code generator.

Or

(b) (i) Consider the following C code segment. Construct the syntax tree representation of the given code and construct the Directed Acyclic Graph (DAG) for the given code : (7)

a = b+ c;

e = a + 1 ;

d = b + c;

f = d + 1;

g = e + f ;

(ii) Consider the basic block given below. Construct the Directed Acyclic Graph (DAG) for the given basic block and find the minimum number of nodes and edges present in the DAG. (6)

u = u + v

v = v + w

x = v – w

y = v – x

z = u + v

15. (a) Explain the various code optimization techniques with example.

Or

(b) Elaborate on the Register allocation techniques.

PART C — (1 × 15 = 15 marks)

16. (a) From the given three address code construct basic block and flow graph. Determine the Next use information and liveness for the variables of each basic block.

1: n = 0
2 : i = 0
3: flag = 0
4: printf("Enter a positive integer: ")
5: scanf("%d", &n)
6: t1 = n = 0
7: t2 = n = =1
8: if False t1 goto 9
9: goto 10
10: flag 1
11: goto 21
12:t3 = i< = n/2
13: if False t3 goto 14
14:t4 = n% i = = 0
15: if True t4 goto 18
16: goto 19
17: break
18: flag = 1
19: i = i + 1
20: goto 12
21: if True flag goto 26
22: printf("%d is a prime number.", n)
23: goto 27
24: goto 27
25: printf("%d is not a prime number.", n)
26: goto 25
27: return 0

Or

(b) A shift-reduce parser carries out the actions specified within braces immediately after reducing with the corresponding rule of grammar.

S → xxW (print "1")

W → y {print "2"}

W → Sz {print "3"}

What is the translation of "xxxxyz" using the syntax-directed translation scheme described by the above rules?