National Institute of Technology Hamirpur
Mid Semester Examination
Computer Science and Engineering
B.Tech. and Dual Degree IIIrd Year, 5th Semester
Subject: Compiler Design (CSD-314)

Max Marks: 30   **Total Time: 90 Min.**

Note: Answer all questions.

1. a. Defined and differentiate a top down and bottom up parsing with the help of suitable example.   (3.0)

b. The statement "When the lexical analyzer read the source-code, it scans the code letter by letter; and when it encounters a whitespace, operator symbol, or special symbols, it decides that a word is completed." is true or false. Write a role of lexical analyzer and Find the number of different tokens in the following segment of code.   (3.0)

```
int Large(int x, int y) {
// This will compare 2 numbers
if (x > y)
return x; }
```

2. a. What do you mean by left factoring? Eliminate left factoring from the following grammar-   (3.0)

$S \rightarrow f \mid fc \mid fcd \mid fcde$

b. What is left recursion? Eliminate left recursion from the following grammar-   (3.0)

exp $\rightarrow$ exp addop term | term
addop $\rightarrow$ + | −
term $\rightarrow$ term mulop factor | factor
mulop $\rightarrow$ ∗
factor $\rightarrow$ (exp) | number

3. a. Calculate the First and Follow sets for the non-terminal symbols of the following grammar then Construct the LL (1) parsing table for the grammar and check whether the grammar is LL(1) or not.   (6.0)

Prog $\rightarrow$ { Stmts } Eof
Stmts $\rightarrow$ Stmt Stmts | $\in$
Stmt $\rightarrow$ id = Expr ; | if (Expr) Stmt
Expr $\rightarrow$ id Etail
Etail $\rightarrow$ + Expr | - Expr | $\in$

4.   Consider the following grammar: (6.0)
S $\rightarrow$ a A d / a c e / b A e
A $\rightarrow$ c
Construct the SLR (1) parsing table for this grammar and check whether the grammar is SLR (1) or not?

5. a. Check the ambiguity of the following grammars and justify your answer.   (3.0)
(i) A $\rightarrow$ AA | (A) | a
(ii) S $\rightarrow$ AA, A $\rightarrow$ aA, A $\rightarrow$ b

b. What do you understand by passes of compiler and Design a two pass compiler for a five programming languages that run on either single machine or all three different machines?   (3.0)