DOF
221416.

SPRING END SEMESTER EXAMINATION-2016

6th Semester B.Tech & B.Tech Dual Degree

COMPILER DESIGN

CS-3002

(Regular-2013 Admitted Batch)

Full Marks: 60 **Time: 3 Hours**

Answer any SIX questions including Question No.1 which is compulsory.

The figures in the margin indicate full marks.

Candidates are required to give their answers in their own words as far as practicable and all parts of a question should be answered at one place only.

1. Answer all the following questions. [2 × 10

a) What is a basic block, explain with an example.

b) The attributes of three arithmetic operators in some programming language are given below (All the operators have their usual meaning and binary in nature)

Operator	Precedence	Associativity
+	High	Left
–	Medium	Right
*	Low	Left

What will be the value of the expression *2-5+1–7*3*?

c) Find all the viable prefixes for the grammar S→aSb|ab.

d) Is the following grammar ambiguous? Justify your answer.

$S \rightarrow aSb|aS|\ \varepsilon$

e) Construct the DAG for the following expression

$$x+x*(y-z)+(y-z)*w$$

f) Define synthesized and inherited attributes.

g) Find out whether the following SDD is L-attributed.

Production	Semantic Rules
$A \rightarrow LM$	$L.i=f(A.i)$; $M.i=f(L.s)$; $A.s=f(M.s)$

h) Explain the following terms
Tokens, Patterns and Lexemes

i) Explain *constant folding* with an example.

j) In the context of compiler design, what do you mean by *reduction in strength*?

2. a) Three address statements of a program are given below [2+2

1. $f=1$
2. $i=2$
3. *if* $(i>x)$ *goto* 9
4. $t1=f*i$
5. $f=t1$
6. $t2=i+1$
7. $i=t2$
8. *goto* 3

(i) Construct the flow graph of the above code

(ii) Identify the loops in your flow graph.

b) Show that the following grammar is $LL(1)$ but not $SLR(1)$. [4

$S \rightarrow AaAb \mid BbBa$
$A \rightarrow \mathcal{E}$
$B \rightarrow \mathcal{E}$

3. a) Translate the statement $x=(a+b) * -c/d$ into [2+2

i) Quadruples

ii) Triples

b) Consider the following Syntax Directed Translation Scheme with non terminals [4

$\{S, A\}$ and terminals $\{a, b\}$

$S \rightarrow aA$ {print 1}
$S \rightarrow a$ {print 2}
$A \rightarrow Sb$ {print 3}

Using the above SDT, write the output printed by a bottom up parser for the input aab.

4. a) Consider the following grammar [4

$S \rightarrow aTA$
$T \rightarrow bT \mid \mathcal{E}$
$A \rightarrow cA \mid d$

Write down the procedures for the non terminals of the grammar to make a recursive descent parser.

b) Show that the following grammar is $LR(1)$ but not $LALR(1)$. [4

$S \rightarrow Aa \mid bAc \mid BC \mid bBa$
$A \rightarrow d$
$B \rightarrow d$

5. a) Construct the DAG for the following basic block and apply *local common sub-expressions* and *dead code eliminations* on it. Assume that a and b are live whereas c and e are not live on exit from the block. [4

$a = b + c$
$b = b - d$
$c = c + d$
$e = b + c$

b) Construct SLR parsing table for the grammar: $S \rightarrow cAd$, $A \rightarrow ab \mid a$. Parse the string *cabd* using the table. [4

6. a) Generate the three-address code for the following code segment: [4

```
int i, x[20];
i = 0;
while( i < 20 )
  {
    x[i] = i ;
      i ++;
    }
```

b) Consider the following SDD. Construct annotated parse tree and dependency graph for the input string 7+8. [4

Production	Semantic Rules
A→BA'	A.inh=B.val A.val=A'.syn
A'→+BA$_1$'	A$_1$'.inh=A'.inh+B.val A'.syn=A$_1$'.syn
A'→ε	A'.syn=A'.inh
B→digit	B.val=digit.lexval

7. a) Explain error recovery strategy in LR parsing with example. [4

b) Construct LL(1) parsing table for the following grammar [4

E→TE'

E'→ +TE'|ε

T→FT'

T'→*FT'|ε

F→(E)|id.

8. Write short notes on <u>any two</u> of the following [4 × 2

a) Symbol Table Management

b) Handle pruning

c) Code generation

– *****