Student Exam No. _____

GANPAT UNIVERSITY

B. Tech SEMESTER VII COMPUTER ENGINEERING

REGULAR EXAMINATION NOV/DEC - 2011

CE 701: COMPILER DESIGN

Time: 3 Hours] **[Total Marks: 70**

Instructions:

1. Figures to the right indicate full marks
2. Each section should be written in a separate answer book
3. Be precise and to the point in your answer

SECTION-I

1. **(A)** What is compiler? State different phase of compiler and discuss synthesis phase with example. [6]

   (B) Check using parser tree that given grammar is ambiguous or not? **S → aSbS | bSaS |** ∈ [4]

   (C) Give advantage and disadvantage of LL (1) parsing. [2]

OR

1. **(A)** What is compiler? State different phase of compiler and explain the function of each phase in brief. [6]

   (B) What is CFG? Write a CFG, which generate palindrome for binary number. [4]

   (C) Remove the Unit production from the following grammar if exist: [2]

   S → AB
   A → B
   B → aB | Bb | ∈

2. **(A)** Consider the following grammar [6]

   S → TS | [S] S |) S | ∈
   T → (X)
   X →T X | [X] X | ∈

   the non-terminal are
   S,T and X and the terminals are (,) ,[and]

   1. Compute the first and follow sets for this grammar
   2. Construct LL(1) parsing table for this grammar
   3. Is this grammar LL (1)?

   (B) Construct the Parse table for given grammar: [5]

   S → ACB | CbB | Ba
   A → da | BC
   B → g | ∈
   C → h | ∈

OR

2. **(A)** What is LL (1) grammar? Find First () and Follow() for the following grammar and check whether grammar is LL(1) or not **[6]**

$S \rightarrow aS \mid Ab$
$A \rightarrow CDE \mid \in$
$C \rightarrow cS \mid \in$
$D \rightarrow dS \mid \in$
$E \rightarrow eS$

(B) Explain following terms with sample CFG: **[5]**

1. Regular grammar
2. Cross Compiler
3. Lexeme
4. Handle
5. Token

3. **(A)** Find Lead() and Last() for the following grammar and construct precedence table for that: **[6]**

$E \rightarrow E + T \mid T$
$T \rightarrow T * F \mid F$
$F \rightarrow (E) \mid id$

(B) What is left recursion in grammar? Explain with an example. **[3]**

(C) Write a CFG for $L = \{ a^{2n} b^{m} / n>=0 , m>=0\}$ **[3]**

SECTION-II

4. **(A)** Consider the following grammar: **[6]**

$E \rightarrow E + T \mid T$
$T \rightarrow T F \mid F$
$F \rightarrow F * \mid a \mid b$

Generate the LR parsing table of the given grammar and check whether the grammar is LR (0) or not?

(B) Differentiate Predictive parser vs. Shift reduce parser **[3]**

(C) Write triples for the expression: **[3]**

(a + b) * (c + d) + (a + b +c)

OR

4. **(A)** Construct LALR parsing table for the following grammar **[6]**

$S \rightarrow AA$
$A \rightarrow aA \mid b$

(B) What is LR parser? How it is different from SLR? **[3]**

(C) Explain top-down and bottom-up parsing. **[3]**

5. **(A)** **[6]**

$E \rightarrow E - T \mid T$
$T \rightarrow F \mid * f$
$F \rightarrow i \mid (E)$

Generate LR parsing table of the given grammar and check whether the grammar is LR (0) or not?

(B) Consider the following code fragment. Generate the **3AC** for it. [5]

```
if a < b then
        while c < d do
            x = x + y
else
        while  e < = f
            q = q + r
```

OR

(A) Explain shift reduce parser and Show the shift-reduce parser action for the string **id1 * (id2 + id3)** for the following grammar. [6]

E → E + E
E → E * E
E → (E)
E → id

(B) Translate the following expression into Quadruples and Triples representations. [5]

A = - B * (C + D) / E

6. (A) Describe the various representations of three-address codes. Translate the expression - **(a + b) * (c + d) + (a + b +c)** into triples and indirect triples. [6]

(B) Consider the following grammar and show the handle of each right sentential form for string **(id + id * id).** [3]

E → E + E
E → E * E
E → id

(C) Differentiate following loop optimization techniques with example [3]
Loop splitting vs. Loop unwinding

----- END OF PAPER -----