Student Exam No._____

GANPAT UNIVERSITY

B. Tech. Semester VII Computer Engineering / Information Technology

Regular Examination November – December 2016

2CE702/2IT702: Compiler Design

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

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

Section I

Que.-1

A Answer the following: **6**

1. Symbol table is necessary for compiler construction, justify your statement with example.
2. Explain the significance of dividing whole compiler functionalities into two parts front-end & back-end.

B Explain in detail the process of compilation. Illustrate the output of each phase of compilation for the input " V = 3.14 * r * r * h " **6**

OR

Que.-1

A Eliminate ϵ production from following grammars: **4**

1. M → ABABABAc | BAC
   A → pA | ϵ
   B → yB | ϵ
   C → x | xx

2. S → ZXCVb | CVXz
   X → aX | ϵ
   Z → b | bZ | ϵ
   V → cV | ϵ
   C → ZV | ϵ

B Remove Left Recursion from below grammars: **4**

1. P → Rp | Qp | q
   Q → pPt | Qrq | ϵ
   R → p | Pt

2. A → CBD | B | a
   B → b | C
   C → A | z | t
   D → d

C What is ambiguity? Check ambiguity for below grammar and check for the acceptance of the string "aabbccdd". **4**

S → AB | C
A → aAb | ab
B → cBd | cd
C → aCd | aDd
D → bDc | bc

Que.-2

A Find First () and Follow () set for the following grammar, construct predictive parsing table and show the parsing steps for string "{p@{qr}r}". **6**

O → p | {OG} | qr
G → @OH | Hs | ϵ
H → #OGt | r | ϵ

B Find First () and Follow () set for the following grammar, construct predictive parsing table and check whether grammar is LL (1) or not? **5**

$S \rightarrow A$
$A \rightarrow aB \mid aC \mid ad \mid ae$
$B \rightarrow bBc \mid f$
$C \rightarrow g$

OR

Que.-2

A Find First() and Follow() for the following grammar and check string "(a(b(2)))(c)" is accepted by this grammar or not? **5**

$M \rightarrow N \mid R$
$N \rightarrow n \mid a \mid b \mid c$
$R \rightarrow (T)$
$T \rightarrow MW$
$W \rightarrow MW \mid \epsilon$

B Find First () and Follow () set for the following grammar, construct predictive parsing table and check whether below grammar is LL (1) or not. **6**

$Q \rightarrow XZY \mid ZbY \mid Ya$
$X \rightarrow da \mid YZ$
$Y \rightarrow g \mid \epsilon$
$Z \rightarrow f \mid \epsilon$

Que.-3

A Write difference between top-down parsing and bottom-up parsing. **3**

B What are the qualities of good compiler? **3**

C Write a code for recursive descent parser for the following grammar and draw a parse tree for string " [0]()() " if accepted by following grammar. **6**

$S \rightarrow TA$
$A \rightarrow (S \mid \epsilon$
$T \rightarrow FB$
$B \rightarrow T \mid \epsilon$
$F \rightarrow PC$
$C \rightarrow)C \mid \epsilon$
$P \rightarrow a \mid f \mid \epsilon \mid [S]$

Section II

Que.-4

A Consider the following grammar: **8**

$S \rightarrow Aa \mid dAb \mid dca \mid cb$

$A \rightarrow c$

Construct SLR (1) parsing table and check given grammar is CLR (1) or not?

B Describe classification of errors in detail. **4**

OR

Que.-4

A Consider the following grammar: **8**

$A \rightarrow A - B \mid B$

$B \rightarrow C \mid *c$

$C \rightarrow (A) \mid a$

Construct LR (0) parsing table and also check whether given grammar is SLR (1) or not?

B Define: Handle. Show the handle for the each step of parsing a string "((b, b, b), b)" for the following grammar: **4**

$X \rightarrow (Y) \mid b$

$Y \rightarrow Y , X \mid X$

Que.-5

A Construct CLR (1) parsing table for the following grammar: **5**

$S \rightarrow A+ B \mid B$

$A \rightarrow \%B \mid \&$

$B \rightarrow A$

B Construct LALR(1) parsing table for the following grammar: **6**

$S \rightarrow A$

$A \rightarrow AB \mid \epsilon$

$B \rightarrow aB \mid b$

OR

Que.-5

A Construct LALR (1) parsing table for the following grammar: **5**

$A \rightarrow BB$

$B \rightarrow bB \mid a$

B Construct SLR (1) parsing table for the following grammar: **6**

$X \rightarrow xXx \mid yXy \mid xy$

Que.-6

A Differentiate Loop splitting and Loop unwinding with example. **4**

B List out various approaches for symbol table organization Explain any two in detail. **4**

C Generate three address code for the following code: **4**

```
1. if p < q then
       while r < s do
             a=a + b
   else
       while d < f do
             b=b + c
```

```
2. main( )
   {
        int x[10], j;
        while (j <=10)
            x[j] = j +1;
   }
```

END OF PAPER