|  |  |  |  |
| --- | --- | --- | --- |
| ACAD-R-42 | SHRI RAMDEOBABA COLLEGE OF ENGINEERING AND MANAGEMENT | | Rev: 00 |
| Department  COMPUTER SCIENCE & ENGINEERING | VII SEMESTER B.E. [COMPUTER SCIENCE & ENGGI.]  SUBJECT: LANGUAGE PROCESSORS  CLASS TEST 1 – EXAMINATION (2014-15)  SHIFT – II (CO1, CO2, CO3, CO4) | | Page: 01/01 |
| Date: 16-08-2014 |
| MAX MARKS: 15 | | TIME: 1 hour (1.30 pm to 2.30pm) | |

**Instructions: Question – 1 (a) is compulsory.**

**Q.1 (a) *Only write answer in the answer sheet. Do not write questions* (0.5x8=4) (CO1, CO2)**

**i.** Name the phases of complier representing Front end and Back end. Reason for having separate phases for Lexical and Syntax analysis.

**ii.** The “€ rule” in FOLLOW SET computation is only used when \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. The FOLLOW SET will never contain \_\_\_\_\_\_\_\_\_\_\_\_\_\_ due to following reason \_\_\_\_\_\_\_\_\_

**iii.** The Left Factoring is not permitted in LL(1) parser due to following reasons: \_\_\_\_\_\_\_\_\_\_\_

**iv.** The only parser used for ambiguous grammar: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ reason\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**v.** Right most derivation in reverse in LR(0) parser means\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

and **A🡪XYZ.** production rule means: **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**vi.** Data Structures used for designing parsers are:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ purpose of data structures: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**vii.** Any two properties of Context Free Grammar: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**viii.** Write Regular expression for language of palindrome string of ODD length with separator.

Q.1 (b) Design Operator Precedence parser for the following grammar. (03)

**E 🡪 E+E | E\*E | id**

Explain the format of production rule in operator precedence grammar. **(CO1, CO4)**

Give the parsing steps for the string: id \* id + id $

**OR**

Q.1 (b) Design the Operator Precedence parser for the following grammar (03)

A 🡪 SR | S

R 🡪 bSR | bS

S 🡪 WbS | W

W🡪 L \* W| L

L 🡪 id

How precedence rule can generated from the given grammar. **(CO1, CO4)**

Design the function table and relation table for the grammar. Comment on space complexity

Q.2 (a) Design the LL(1) Parser for the grammar. Why Left Recursion is not permitted in top down parsing. (04)

Grammar:

S 🡪 cBuET | b **(CO1, CO4)**

B 🡪 BS | c

E 🡪 b

T 🡪 dBe | €

Q.2 (b) Design LR(0) parser for the grammar. Why left recursion is permitted in bottom up parsing.

Grammar: S 🡪 Aa | bAc | Bc (04)

A 🡪 d **(CO3, CO4)**

B 🡪 d

Draw the DFA.

**OR**

Q.3 (a) Design LL(1) parser for the grammar. If it is required to modify parser to perform LL(2) parsing, suggest modifications. (04)

S 🡪 B C c | g D B **(CO1, CO4)**

B 🡪 b C D E | €

C 🡪 D a B | c a

D 🡪 € | d D

E 🡪 E a f | €

Q.3 (b) Design LR(0) Parser. What is “L”, “R”, and “0”, explain with suitable example. (04)

**S 🡪 AA (CO3, CO4)**

**A 🡪 aA | b**

Draw the DFA.