# GUJARAT TECHNOLOGICAL UNIVERSITY

**BE - SEMESTER–VII(NEW)   EXAMINATION – SUMMER 2019**

**Subject Code: 2170701**                                      **Date:10/05/2019**

**Subject Name: Compiler Design**

**Time: 02:30 PM TO 05:00 PM**                          **Total Marks: 70**

**Instructions:**

1. **Attempt all questions.**
2. **Make suitable assumptions wherever necessary.**
3. **Figures to the right indicate full marks.**

|  |  |  | MARKS |
|---|---|---|---|
| **Q.1** | **(a)** | Define lexemes, patterns and tokens. | 03 |
|  | **(b)** | Differentiate compilers and interpreters. | 04 |
|  | **(c)** | Explain analysis of source program for compilers. | 07 |
| **Q.2** | **(a)** | Give regular definition for signed and unsigned numbers. | 03 |
|  | **(b)** | Check whether the following grammar is ambiguous or not. | 04 |

S→ (S) S
S→ ε

**(c)** Draw DFA from regular expression without constructing NFA.      **07**
(a | b | c)* a (b | c)* #

**OR**

**(c)** Draw NFA from regular expression using Thomson's construction and convert it into DFA.      **07**
(a | b)* a b* a

**Q.3 (a)** Define handle and handle pruning.      **03**

**(b)** Construct operator precedence relations table for following grammar.      **04**
E→ E+E
E→ E-E
E→ E*E
E→ (E)
E→ id
Assume suitable operator associativity and precedence.

**(c)** Construct recursive descent parser for following grammar.      **07**
E→ T A
A→ + T A
A→ ε
T→ F B
B→ * F B
B→ ε
F→ ( E )
F→ id

**OR**

**Q.3 (a)** Differentiate top down parsing and bottom up parsing.      **03**

**(b)** Construct syntax directed translation scheme for infix to postfix conversion.      **04**

**(c)** Construct LL(1) parsing table for following grammar. Check whether the grammar is LL(1) or not.      **07**
A → A a B
A → x
B → B C b
B → C y
C → C c
C → ε

1

| Q.4 | (a) | Define Intermediate code and its importance. | 03 |
| | (b) | Construct LR(0) item sets for following grammar. | 04 |

S → AaAb
S → BbBa
A → ε
B → ε

| | (c) | Explain various error recovery schemes in detail. | 07 |

**OR**

| Q.4 | (a) | Differentiate LR(1) and LALR(1) parsers. | 03 |
| | (b) | Construct syntax tree and DAG for following expression. | 04 |

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

| | (c) | Explain quadruples, triples and indirect triples with examples. | 07 |

| Q.5 | (a) | Define basic block with a simple example. | 03 |
| | (b) | Explain activation record. | 04 |
| | (c) | Explain various methods of peephole optimization. | 07 |

**OR**

| Q.5 | (a) | Explain static storage allocation. | 03 |
| | (b) | Explain any two parameter passing methods. | 04 |
| | (c) | Explain various issues in design of code generator. | 07 |

*************

2