BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
HYDERABAD CAMPUS
SECOND SEMESTER 2021-22

COMPILER CONSTRUCTION (CS F363)
Programming Assignment – 2

LAST DATE: 24/4/2022 **MAX MARKS: 20M**

Requirement for Parser and Intermediate code generator

Phase-2 (Requirement for Parser and Intermediate Code Generator Deliverables, Test Suite, How-to-approach Phase-2)

• Requirements Specification:

o Input: Lexer generated tokens
o Output1: Parse Tree in the form of any tree traversal or level wise output of the nonterminal from left to right. In case of errors your parser must report the errors and continue parsing.
o Output2: Intermediate code in the form of quadruples for program construct expressions and conditional statements.

• Files
o Interface file : parser.h
o Implementation : parser.c
o Implementation : Intermediate.c

Stage-2 How-to

1. Read the language Specification: the overview, the grammar (natural form) and the tokens to gain an over-all understanding.
2. Apply your understanding to the given examples and work out the details by deriving the program text using the rules given in the grammar.
3. Understand any type of parser for your grammar.
4. Use http://jsmachines.sourceforge.net/machines/ and select the parser you would like ot implement and paste your CFG to get the parse table.
5. Using the parse table as input, implement the parser as discussed in the class
6. Test the scanner with the test cases.
7. Write a procedure to traverse and print the parse tree or the sequence of tokens from the stack.
8. Implement SDT/SDD to generate the Intermediate code for arithmetic expressions and conditional statements in your language.
9. Write your own test cases and document your code.

Rubric for evaluation

1. Parsing algorithm and printing the parse tree or stack contents – 5
2. On error the parser must not stop but continue -2.5
3. A few Error handling routines -2.5
4. SDD/SDT implementation -5
5. Viva – 5