**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI**

Batch No. :

**DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS**

**Compiler Construction (CS F363)**

**II Semester 2017-18**

**Compiler Project (Stage-2 Submission)**

**Coding Details**

**(April 20, 2018)**

*Instruction: Write the details precisely and neatly. Places where you do not have anything to mention, please write NA for Not Applicable.*

1. ID Number: 2015A7PS0040P

Name: Rohit Lodha\_\_\_\_\_\_

1. Mention the names of the Submitted files ( Include Stage-1 and Stage-2 both)

1\_lexer.c\_ 7\_\_\_\_\_ast.h\_ 13\_\_\_ast.c\_\_\_\_ 19\_\_\_ testcase4.txt \_\_\_

2\_lexer.h\_\_ 8\_\_\_astDef.h\_\_ 14\_\_\_ semantics.h\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ testcase5.txt \_\_\_

3\_\_lexerDef.h\_ 9\_\_\_symbolTable.c\_\_ 15\_\_ semanticsDef.h\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 21\_\_\_ testcase6.txt \_

4\_\_parser.c\_\_ 10\_\_symbolTable.h\_\_ 16\_\_testcase1.txt\_\_\_\_\_\_\_\_ 22\_\_\_\_c1.txt

5\_\_parser.h\_\_\_ 11\_\_symbolTableDef.h\_ 17\_ testcase2.txt \_ 23\_\_ c2.txt

6\_\_parserDef.h\_\_\_\_\_\_\_\_ 12\_\_semantics.c\_\_ 18\_\_ testcase3.txt \_\_\_ 24\_\_ c3.txt

25\_\_grammer.txt 26\_\_\_driver.c 27\_\_\_makefile 28\_codingDetails.docx

1. Total number of submitted files: \_28\_ (All files should be in ONE folder named exactly as your ID)
2. Have you compressed the folder as specified in the submission guidelines? (yes/no)\_\_\_Yes\_\_
3. **Status of Code development**: Mention 'Yes' if you have developed the code for the given module, else mention 'No'.
   1. Lexer (Yes/No): \_\_\_\_\_\_Yes\_\_\_\_\_
   2. Parser (Yes/No):\_\_\_\_\_\_\_Yes\_\_\_\_\_\_\_\_\_\_\_
   3. Abstract Syntax tree (Yes/No):\_\_\_\_Yes\_\_\_\_\_
   4. Symbol Table (Yes/ No):\_\_\_\_\_\_Yes\_\_\_\_\_\_
   5. Type checking Module (Yes/No):\_\_\_\_\_Yes\_\_\_\_(implemented in semantics.c)\_\_
   6. Semantic Analysis Module (Yes/ no):\_\_Yes\_\_\_(reached LEVEL \_\_4\_\_ as per the details uploaded)
   7. Code Generator (Yes/No):\_\_\_\_\_NO\_\_
4. **Execution Status**:
   1. Code generator produces code.asm (Yes/ No):\_\_\_\_No\_\_\_
   2. code.asm produces correct output using NASM for testcases (C#.txt, #:1-3):\_\_\_No\_\_\_\_\_\_\_\_\_
   3. Semantic Analyzer produces semantic errors appropriately (Yes/No):\_\_Yes\_\_\_
   4. Type Checker reports type mismatch errors appropriately (Yes/ No):\_\_\_Yes\_\_\_\_\_
   5. Symbol Table is constructed (yes/no)\_\_Yes\_and printed appropriately (Yes /No):\_\_\_\_\_Yes\_\_\_\_
   6. AST is constructed (yes/ no) \_\_\_Yes\_\_and printed (yes/no) \_\_\_Yes\_
   7. Name the test cases out of 9 as uploaded on the course website for which you get the segmentation fault (testcase#.txt ; # 1-6 and c@.txt ; @:1-3):\_\_\_\_\_None(All pass the semantic stage)\_
5. **Data Structures** (Describe in maximum 2 lines and avoid giving C definition of it)
   1. AST node structure\_\_: Include all the structure fields of parse tree along with a pointer to current symbol table to be looked at for variables
   2. Symbol Table structure:\_ Tree structure with each node containing pointers to function Table and Variable Table. ROOT Table is the global table( Head of Linked List) and function/if/else are the other nodes.(which defines a specific block of code)
   3. Matrix type expression structure: Check for <matrix> rulenode in the ast tree and process accordingly to update/check the symbol table(for size of rows and columns) and check semantic with other operators
   4. Input parameters type structure:\_\_\_Linked List of variable\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   5. Output parameters type structure:\_\_\_Linked List of variable\_\_\_\_\_
   6. Structure for maintaining the three address code(if created) :\_\_None\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
6. **Semantic Checks:** Mention your scheme NEATLY for testing the following major checks (in not more than 5-10 words)[ Hint: You can use simple phrases such as 'symbol table entry empty', 'symbol table entry already found populated', 'traversal of linked list of parameters and respective types' etc.]
   1. Variable not Declared :\_Symbol Table entry not found
   2. Multiple declarations: \_Symbol table entry already found populated
   3. Number and type of input and output parameters: Linked List matching of both types of parameters and count of parameters
   4. assignment of value to the output parameter in a function: Type of expression result of righthand side is matched with the declared type of lefthand side
   5. function call semantics:\_Linked List matching of both types of parameters and count of parameters
   6. type checking :\_Type of expression on both sides should match
   7. return semantics:\_Variable assigned a value or not in the function defination
   8. Recursion :\_Symbol Table entry checked in the parent symbol Table
   9. module overloading:\_\_ Symbol Table entry checked
   10. 'If' semantics :\_\_Boolean expression check for correctness and a new block of symbol table created for variables declared in IF/ELSE block\_\_\_\_\_
   11. Matrix semantics and type checking of matrix type variables: While creating symbol table, whenever a matrix is identified at the LHS of some assignment, its size of row and col are updated by calculating from the RHS, type checking is done both on the basis of size of rows and columns in PLUS and MINUS operator, in MUL and DIV – matrix type is invalid
   12. register allocation (your manually selected heuristic) : NA
   13. Scope of variables and their visibility: Symbol Table points to the current scope Symbol table for each variable inside the function/if/else. Checked with parent symbol table too.
7. **Compilation Details**:
   1. Makefile works (yes/No):\_\_Yes
   2. Code Compiles (Yes/ No):\_\_\_Yes\_
   3. Mention the .c files that do not compile:\_\_\_\_N/A
   4. Any specific function that does not compile:\_\_\_N/A
   5. Ensured the compatibility of your code with the specified gcc version(yes/no)\_\_\_Yes
8. **Driver Details**: Does it take care of the options specified earlier?(yes/no):\_\_\_\_Yes
9. Specify the language features your compiler is not able to handle (in maximum one line):

Code Generation

1. Are you availing the lifeline (Yes/No): \_\_\_\_No\_\_
2. Write exact command you expect to be used for executing the code.asm using NASM simulator [We will use these directly while evaluating your NASM created code]

\_\_\_\_NA\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Strength of your code**(Strike off where not applicable): (a) correctness (b) completeness (c) robustness ~~(d) Well documented~~ (e) readable (f) strong data structure (f) Good programming style (indentation, avoidance of goto stmts etc) (g) modular (h) space and time efficient
2. Any other point you wish to mention: \_NA\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Declaration:** I, Rohit Lodha (your name) declare that I have put my genuine efforts in creating the compiler project code and have submitted the code developed by me. I have not copied any piece of code from any source. If my code is found plagiarized in any form or degree, I understand that a disciplinary action as per the institute rules will be taken against me and I will accept the penalty as decided by the department of Computer Science and Information Systems, BITS, Pilani.

Sign:\_\_Rohit\_

ID\_\_\_2015A7PS0040P\_

Name:\_\_Rohit Lodha\_\_\_\_\_\_

Date: \_20/04/18\_

-------------------------------------------------------------------------------------------------------------------------------------------------

/\*not to exceed three pages\*/