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

**DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS**

**Compiler Construction (CS F363)**

**II Semester 2019-20**

**Compiler Project (Stage-1 Submission)**

**Coding Details**

**Group No.**

**6**

**(February 24, 2020)**

1. IDs and Names of team members

*ID: 2016B4A70930P Name: PRASHANT KHANDELWAL*

*ID: 2016B4A70891P Name: NAMAN DEEP SRIVASTAVA*

*ID: 2016B2A70901P Name: RACHIT AGRAWAL*

*ID: 2016B1A70870P Name: MOHIT KRIPLANI*

*ID: 2016B3A70581P Name: UTKARSH AGARWAL*

1. Mention the names of the Submitted files :

*1. driver.c 2. lexer.c 3. parser.c*

*4. hashTable.c 5. helper.c 6. lexer.h*

*7. parser.h 8. hashTable.h 9. helper.h*

*10. lexerDef.h 11. parserDef.h 12. keywords.txt*

*13. Updated\_Grammar.txt 14. makefile 15. constants.h*

*16. Coding Details Proforma 17. testcases (6 text files)*

1. Total number of submitted files: *22*

(All files should be in **ONE folder** named exactly as Group\_#, # is your group number)

1. Have you mentioned your names and IDs at the top of each file (and commented well)? (Yes/ no) *Yes*

[Note: Files without names will not be evaluated]

1. Have you compressed the folder as specified in the submission guidelines? (yes/no) *Yes*
2. **Lexer Details:**
   1. Technique used for pattern matching: *Traversal of DFA*
   2. DFA implementation (State transition using switch case, graph, transition table, any other (specify): *State transition using switch case*
   3. Keyword Handling Technique: *Hashing*
   4. Hash function description, if used for keyword handling: *Quadratic Probing*
   5. Have you used twin buffer? (yes/ no): *No*
   6. Lexical error handling and reporting (yes/No): *Yes*
   7. Describe the lexical errors handled by you: *Identifier longer than the prescribed length, unknown symbol, unknown pattern*
   8. Data Structure Description for tokenInfo (in maximum two lines): *A struct type having token name, token value, line number and pointer to next token.*
   9. Interface with parser: *getNextToken*
3. **Parser Details:** 
   1. **High Level Data Structure Description (in maximum three lines each, avoid giving C definitions used):**
      1. grammar: *Array of grammar rules where each rule is a linked-list having first node as the non-terminal (antecedent of the grammar rule) and rest contains terminals / non-terminals in the consequent of the rule in order. Each node also contains the following information: non-terminal / terminal, hash value and pointer to the next node .*
      2. parse table: *Double-dimensional integer array where rows are non-terminal hash value and columns are terminal hash value and values are rule numbers.*
      3. parse tree: (Describe the node structure also): *A struct type having hash value, pointer to parent, child and next, if it is non-terminal and token pointer.*
      4. Parsing Stack node structure: *Array of parse tree node.*
      5. Any other (specify and describe): *No*
   2. **Parse tree** 
      1. Constructed (yes/no): *Yes*
      2. Printing as per the given format (yes/no): *Yes*
      3. Describe the order you have adopted for printing the parse tree nodes (in maximum two lines)

*Inorder Traversal*

*Leftmost child  Parent node  remaining siblings (excluding the leftmost child)*

* 1. **Grammar and Computation of First and Follow Sets** 
     1. Data structure for original grammar rules: *Array of grammar rules where each rule is a linked-list having first node as the non-terminal (antecedent of the grammar rule) and rest contains terminals / non-terminals in the consequent of the rule in order. Each node also contains the following information: non-terminal / terminal, hash value and pointer to the next node .*
     2. FIRST and FOLLOW sets computation automated (yes /no): *Yes*
     3. Data structure for representing sets: *Double-dimensional integer array where each row corresponds to hash value of non-terminals that stores array of hash values of terminals .*
     4. Time complexity of computing FIRST sets: *O(nmr2)*

*Where n -> no. of non-terminals, m -> no. of terminals and r -> no. of rules*

* + 1. Name the functions (if automated) for computation of First and Follow sets

*findFirstSet – computes first sets*

*findFollowSet – computes follow sets*

*ComputeFirstAndFollowSets – calls the above two functions*

* + 1. If computed First and Follow sets manually and represented in file/function (name that): *No*
  1. **Error Handling** 
     1. Attempted (yes/ no): *Yes*
     2. Printing errors (All errors/ one at a time) : *All errors*
     3. Describe the types of errors handled

*ERROR\_1: l**line\_number: Identifier is longer than the prescribed length*

*ERROR\_2: line\_number: Unknown Symbol*

*ERROR\_3: line\_number: Unknown pattern*

*ERROR\_4: Unable to hash*

*ERROR\_5: Line line\_number – Error in the the token act\_tkn for lexeme as expected token is exp\_tkn*

* + 1. Synchronizing tokens for error recovery (describe)

*First-Follow hybrid using Panic mode error recovery*

* + 1. Total number of errors detected in the given testcase t6(with\_syntax\_errors).txt:

*2 lexical and 10 syntactic errors (All specified in t6.txt)*

1. **Compilation Details:**
   1. Makefile works (yes/no): *Yes*
   2. Code Compiles (yes/ no): *Yes*
   3. Mention the .c files that do not compile: *None*
   4. Any specific function that does not compile: *None*
   5. Ensured the compatibility of your code with the specified gcc version(yes/no): *Yes*
2. **Driver Details**: Does it take care of the options specified earlier(yes/no): *Yes*
3. **Execution** 
   1. status (describe in maximum 2 lines): *Working for all testcases provided.*
   2. Execution time taken for
      * t1.txt (in ticks): L - 648, P - 4810 and (in seconds): L - 0.000648, P – 0.004810
      * t1.txt (in ticks): L - 518, P - 8126 and (in seconds): L - 0.000518, P – 0.008126
      * t3.txt (in ticks): L - 539, P - 9465 and (in seconds): L - 0.000539, P – 0.009465
      * t4.txt (in ticks): L - 725, P - 9917 and (in seconds): L - 0.000725, P – 0.009917
      * t5.txt (in ticks): L - 804, P - 8722 and (in seconds): L - 0.000804, P – 0.008722
      * t6.txt (in ticks): L - 936, P - 9767 and (in seconds): L - 0.000936, P – 0.009767
   3. Gives segmentation fault with any of the test cases (1-6) uploaded on the course page. If yes, specify the testcase file name: *No*
4. Specify the language features your lexer or parser is not able to handle (in maximum one line)

*None that we know of.*

1. Are you availing the lifeline (Yes/No): *Yes*
2. Declaration: We, Prashant, Naman, Rachit, Mohit and Utkarsh declare that we have put our genuine efforts in creating the compiler project code and have submitted the code developed only by our group. We have not copied any piece of code from any source. If our code is found plagiarized in any form or degree, we understand that a disciplinary action as per the institute rules will be taken against us and we will accept the penalty as decided by the department of Computer Science and Information Systems, BITS, Pilani.

ID: 2016B4A70930P Name: Prashant Khandelwal

ID: 2016B4A70891P Name: Naman Deep Srivastava

ID: 2016B2A70901P Name: Rachit Agrawal

ID: 2016B1A70870P Name: Mohit Kriplani

ID: 2016B3A70581P Name: Utkarsh Agarwal

Date: 24-02-2020

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

Should not exceed 4 pages.