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

**DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS**

**Compiler Construction (CS F363)**

**II Semester 2017-18**

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

**Coding Details**

**(February 26, 2018)**

1. **Personal details**

ID: 2014B2A70201P

Name: Aman Gupta

1. **Files and folder details**
2. Mention the names of the Submitted files :

1) codingDetails.docx 7) lexer.h 13) parserDef.h

2) driver.c 8) lexerDef.h 14) testcase1.txt

3) first.txt 9) makefile 15) testcase2.txt

4) firstAndFollow.txt 10) modifiedGrammar.txt 16) testcase3.txt

5) follow.txt 11) parser.c 17) testcase4.txt

6) lexer.c 12) parser.h 18) testcase5.txt

1. Total number of submitted files: 18 (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. **Lexer Details:**
   1. Technique used for pattern matching: Implemented Submitted DFA using nested switch-case. The outer switch works on the lookahead whereas the inner switch works on the state of the DFA.
   2. Keyword Handling Technique: Hash Table
   3. Hash function description, if used for keyword handling: (Sum of ASCII values of the ID/FUNID)%23
   4. Have you used twin buffer? (yes/ no) Yes
   5. Error handling and reporting (yes/No): Yes
   6. Describe the errors handled by you:
      1. Unknown Patterns
      2. Length Restrictions
      3. Unknown Symbol
   7. Data Structure Description for tokenInfo (in maximum two lines): A character array of size 21 to store lexeme, an integer variable to store the token class & an unsigned long long to store line number of the token
4. **Parser Details:** 
   1. High Level Data Structure Description (in maximum three lines each, avoid giving C definitions used):
      1. grammar: An array (of size = No. of Rules) of pointers to RHS of the rule which are stored in the form of linked lists.
      2. parse table: A 2-D array of pointers (of size = NO\_OF\_NON\_TERMINALS \* (NO\_OF\_TERMINALS + 1)) towards first element of RHS of Rule stored in the grammar data structure
      3. parse tree: (Describe the node structure also): A N-ary tree of nodes consists of tokenInfo, pointer to parent, pointer to firstChild and a pointer to the nextSibling.
      4. Any other (specify and describe) - Stack: Created using Linked List implementation. All relevant functions implemeted as well.
   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
   3. Computation of First and Follow Sets
      1. Data structure for First and Follow sets: Same as Grammar
      2. FIRST and FOLLOW sets computation automated (yes /no): No
      3. Name the functions (if automated) for computation of First and Follow sets\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
      4. If computed First and Follow sets manually and represented in file/function (name that): First Sets of Non-Terminals (first.txt), Follow Sets of Non-Terminals (follow.txt), rule-wise first and follow sets (firstAndFollow.txt) in which if a rule had EPSILON in its first set, the EPSILON was relaced with follow set of LHS.
   4. Error Handling and recovery
      1. Attempted (yes/ no): Handling: Yes (Generated Partial Parse Tree in case of Error), Recovery: No
      2. Synchronizing set formation details:
      3. Describe the types of errors handled
         1. Rule not existing corresponding to a non-terminal on topOfStack and a terminal as lookahead.
         2. topOfStack and lookahead token mismatch
         3. topOfStack is DOLLAR but there is still content in the file to be read
         4. EndOfFile Reached but topOfStack is not DOLLAR
5. **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
6. **Driver Details:** Does it take care of the options specified earlier(yes/no): Yes
7. **Execution details**
   1. status (describe in maximum 2 lines): Code compiles and executes successfully on all 5 given testcases without any segmentation fault.
   2. Gives segmentation fault with any of the revised test cases (1-5) uploaded on the course page. If yes, specify the testcase file name: None
8. Specify the language features your lexer or parser is not able to handle (in maximum one line): None
9. **Lifeline detail:** Are you availing the lifeline (Yes/No): No
10. **Declaration**: I, Aman Gupta (your name) declare that I have put my genuine efforts in creating the compiler project code and have submitted the code developed only 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.

ID: 2014B2A70201P

Name: Aman Gupta

Date: 26th February, 2018

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

/\*not to exceed two pages\*/

**NOTE:** On selecting option 3, the parse tree is generated in written to a file (parsetreeoutfile.txt assuming the following command is executed in the terminal - $./stage1exe testcase.txt parsetreeoutfile.txt).

On selecting option 4, the parse tree is read from the above mentioned file and printed to the console.