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

2015

**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 2015A7PS0127P

Name SAMIP JASANI

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

1 \_Stack.c 7 codingDetails.docx 13 makefile

2 \_Stack.h 8 driver.c 14 parser.c

3 \_Tree.c 9 Grammer.txt 15 parser.h

4 \_Tree.h 10 lexer.c 16 parserDef.h

5 \_Trie.c 11 lexer.h

6 \_Trie.h 12 lexerDef.h

1. Total number of submitted files: 16 (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: DFA implemented using switch-case and if-else constructs
   2. Keyword Handling Technique: used Trie for checking if ID is a keyword used strcmp incase of checking “\_main”
   3. Hash function description, if used for keyword handling: N/A
   4. Have you used twin buffer? (yes/ no): no
   5. Error handling and reporting (yes/No): yes
   6. Describe the errors handled by you : Error handled for unknown symbol, unknown pattern, Identifier longer than prescribed length, String longer than prescribed length.
   7. Data Structure Description for tokenInfo (in maximum two lines): I am using a struct token for tokenInfo. It contains enum for typeofToken, int for lineno, float for value in case of NUM or RNUM, char\* for string read.
4. **Parser Details:** 
   1. High Level Data Structure Description (in maximum three lines each, avoid giving C definitions used):
      1. grammar : Grammer is array of Rules. Each Entry in Grammer corresponds to a non- terminal. Rules are nodes containing pointer to rhside of rules and next rule. Rhside are node containing typeinfpo , terminal detail and next rhside component
      2. parse table : ParseTable is 2D array of rules where 1st Dimension is for non-terminal and 2nd for terminals. It can have atmost one rule. If it is NULL then there is no rule for corresponding entry.
      3. parse tree: (Describe the node structure also) :It is n-ary tree built corresponding to the source code. Each node has detail about its leaf detail, contains pointer to either token (in case of terminal) or rhside (in case of non-terminal), it also have pointers to its child, sibling and parent;
      4. stack: made a stack using array for PDA
      5. rules: linklist for storing multiples grammer rules
      6. rhside: linklist for rule’s rhside components
   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)

Parse tree are printed in inorder fashion. As our tree is n-ary Each node will print its child first then itself and then all siblings of its child.

* 1. Computation of First and Follow Sets
     1. Data structure for First and Follow sets : both uses a char\*\* where 1st dimension is non-terminal and 2nd is terminal
     2. FIRST and FOLLOW sets computation automated (yes /no): yes
     3. Name the functions (if automated): for computation of First and Follow sets: MakeFirst to construct First Set. MakeFollow to construct Follow Set.
     4. If computed First and Follow sets manually and represented in file/function (name that) : N/A
  2. Error Handling and recovery
     1. Attempted (yes/ no): yes
     2. Synchronizing set formation details : Synchronizing over SEMICOLON and removing Terminals from stack in case of mixmatch.
     3. Describe the types of errors handled : Syntax error in each line is reported and parser moves to next line.

1. **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
2. **Driver Details:** Does it take care of the options specified earlier(yes/no): yes
3. **Execution details**
   1. status (describe in maximum 2 lines): Every thing works with out gcc errors
   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: N\A
4. Specify the language features your lexer or parser is not able to handle (in maximum one line) Almost everything is handled
5. **Lifeline detail:** Are you availing the lifeline (Yes/No): No
6. **Declaration**: I, SAMIP JASANI (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: 2015A7PS0127P

Name: SAMIP JASANI

Date: 26/02/2018

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

/\*not to exceed two pages\*/