

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
(February 24, 2020)

Group No.

31

1. IDs and Names of team members

ID: 2016B4A70933P Name: MADHUR PANWAR

ID: 2016B3A70528P Name: TUSSANK GUPTA

ID: 2016B4A70580P Name: SALMAAN SHAHID

ID: 2016B3A70549P Name: APURV BAJAJ

ID: 2016B5A70452P Name: HASAN NAQVI

2. Mention the names of the Submitted files :

1	lexer.h	11	generic_stack.c	21	util.c
2	lexer.c	12	generic_stack.h	22	util.h
3	lexerDef.h	13	treeNodePtr_stack.c	23	grammar.txt
4	parser.c	14	treeNodePtr_stack.h	24	gr_hr.txt
5	parser.h	15	errorPtr_stack.c	25	keywords.txt
6	parserDef.h	16	errorPtr_stack.h	26	nonTerminals.txt
7	config.h	17	set.c	27	tokens.txt
8	error.c	18	set.h	28	makefile
9	error.h	19	hash.c	29	coding details stage1.pdf
10	driver.c	20	hash.h		

Test Case files

30	t1.txt	34	t5.txt	38	t9.txt
31	t2.txt	35	t6.txt	39	t10.txt
32	t3.txt	36	t7.txt		
33	t4.txt	37	t8.txt		

3. Total number of submitted files: 39 (All files should be in **ONE folder** named exactly as Group_#, # is your group number)
4. 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]
5. Have you compressed the folder as specified in the submission guidelines? (yes/no) Yes

6. **Lexer Details:**

- [A]. Technique used for pattern matching: DFA
- [B]. DFA implementation (State transition using switch case, graph, transition table, any other (specify): Switch Case with each case representing a state of the DFA
- [C]. Keyword Handling Technique: Hash Table and ENUM filled using keywords.txt
- [D]. Hash function description, if used for keyword handling: djb2 Hash Function which uses Horner's rule for hash computation.
- [E]. Have you used twin buffer? (yes/ no) Yes
- [F]. Lexical error handling and reporting (yes/No): Yes
- [G]. Describe the lexical errors handled by you All types of lexical errors
- [H]. Data Structure Description for tokenInfo (in maximum two lines):
struct tokenInfo { tokenType type; unsigned int lno; char lexeme[101]; union { int num; float rnum; } value; };
Tagged union (tagged with enum tokenType type) within a struct tokenInfo
- [I]. Interface with parser tokenInfo* getNextToken(FILE *file_ptr)

7. **Parser Details:**

[A]. **High Level Data Structure Description (in maximum three lines each, avoid giving C definitions used):**

- i. **grammar** : An array of grammar nodes with each node containing a left hand side symbol and a head to a linked list containing right hand side symbols of that rule.
- ii. **parse table** : A two dimensional integer array having number of rows equal to the number of non terminals and number of columns equal to the number of terminals (including \$ symbol)
- iii. **parse tree** (Describe the node structure also) : A linked list based tree of treeNodes with each node containing a token symbol, a tokenInfo returned by lexer (only for terminals), a next pointer to its next sibling, a child pointer to its first child and a parent pointer to its parent.
- iv. **Parsing Stack node structure** : Each stack node contains the pointer to the node of parse tree and a next pointer which connects it to the stack node below it.
- v. **Any other (specify and describe)** : In addition to the above, we have an error stack which stores the errors occurring across all compilation phases. Its node contains a pointer to a structure containing error data and a next pointer which connects it to the stack node below it (errors are pushed to the stack in their order of occurrence).

[B]. **Parse tree**

- i. Constructed (yes/no): ___Yes___
- ii. Printing as per the given format (yes/no): ___Yes___
- iii. Describe the order you have adopted for printing the parse tree nodes (in maximum two lines)

Inorder traversal is used to print the n-ary parse tree. A node is printed only when its leftmost child has been printed inorder (recursive call) and then each of its remaining children are printed inorder (recursive calls to each child).

[C]. **Grammar and Computation of First and Follow Sets**

- i. **Data structure for original grammar rules** _Array of a structure containing symbol of left hand side non-terminal and head pointers to linked lists representing right hand sides of productions._
- ii. **FIRST and FOLLOW sets computation automated (yes /no)** _____Yes _____
- iii. **Data structure for representing sets** _____*unsigned long long int*_____
- iv. **Time complexity of computing FIRST sets :**
___ $O(|GrammarRules| * \max\{|rhsOfProduction|\} * |Terminals|)$ ___[Assuming constant time for set union, implemented as *bitwiseOR* of two *unsigned long long ints*.]_____
- v. **Name the functions (if automated) for computation of First and Follow sets:**
_____ *populateFirstSet()* and *populateFollowSet()* respectively. _____
- vi. **If computed First and Follow sets manually and represented in file/function (name that)** _N.A._

[D]. **Error Handling**

- i. **Attempted (yes/ no):** ___Yes___
- ii. **Printing errors (All errors/ one at a time) :** _____All Errors _____
- iii. **Describe the types of errors handled**
 - 1. Lexical Error when the lexeme does not conform to any of the valid patterns.
 - 2. Syntax Error when the non-terminal on top of the stack cannot produce the token in the input stream.
 - 3. Syntax Error when the terminal on the top of the stack does not match the token in the input stream.
 - 4. Syntax Error when the parsing stack is non-empty but input stream has been read completely.
- iv. **Synchronizing tokens for error recovery (describe):** _For every non-terminal, the synchronizing set is constructed by *union of a default set of tokens* (common for all non-terminals) *and follow set of that non-terminal*. _____
- v. **Total number of errors detected in the given testcase t6(with_syntax_errors).txt :**
_____All errors: 2 Lexical Errors and 10 Syntax Errors_____

8. Compilation Details:

[A]. Makefile works (yes/no): ____ Yes ____

[B]. Code Compiles (yes/ no): ____ Yes ____

[C]. Mention the .c files that do not compile: ____ N.A. ____

[D]. Any specific function that does not compile: ____ N.A. ____

[E]. Ensured the compatibility of your code with the specified gcc version(yes/no) ____ Yes ____

9. Driver Details: Does it take care of the options specified earlier(yes/no): ____ Yes ____

10. Execution

[A]. status (describe in maximum 2 lines): __ The complete code, post build, executes as expected, displaying the driver menu and performs in desired manner depending on user input. ____

[B]. Execution time taken for

- t1.txt (in ticks) ____ 209.00 ____ and (in seconds) ____ 0.000209 ____
- t2.txt (in ticks) ____ 115.00 ____ and (in seconds) ____ 0.000115 ____
- t3.txt (in ticks) ____ 179.00 ____ and (in seconds) ____ 0.000179 ____
- t4.txt (in ticks) ____ 365.00 ____ and (in seconds) ____ 0.000365 ____
- t5.txt (in ticks) ____ 419.00 ____ and (in seconds) ____ 0.000419 ____
- t6.txt (in ticks) ____ 469.00 ____ and (in seconds) ____ 0.000469 ____

[C]. Gives segmentation fault with any of the test cases (1-6) uploaded on the course page. If yes, specify the testcase file name: ____ No Segmentation fault ____

11. Specify the language features your lexer or parser is not able to handle (in maximum one line) ____ N.A. ____

12. Are you availing the lifeline (Yes/No): ____ No ____

13. Declaration: We, **MADHUR PANWAR, TUSSANK GUPTA, SALMAAN SHAHID, APURV BAJAJ** and **HASAN NAQVI** 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. [Write your ID and names below]

ID: <u>2016B4A70933P</u>	Name: <u>MADHUR PANWAR</u>
ID: <u>2016B3A70528P</u>	Name: <u>TUSSANK GUPTA</u>
ID: <u>2016B4A70580P</u>	Name: <u>SALMAAN SHAHID</u>
ID: <u>2016B3A70549P</u>	Name: <u>APURV BAJAJ</u>
ID: <u>2016B5A70452P</u>	Name: <u>HASAN NAQVI</u>

Date: ____ 24/02/2020 ____