BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS

Group No. 22

Compiler Construction (CS F363)
II Semester 2022-23
Compiler Project (Stage-1 Submission)
Coding Details
(March 2, 2023)

---

1. IDs and Names of team members

ID: 2020A7PS0048P Name: Mohit Makwana

ID: 2020A7PS0058P Name: Kathan Patel

ID: 2020A7PS0299P Name: Samay Gandhi

ID: 2020A7PS1511P Name: Aditya Sheth

ID: 2020A7PS1692P Name: Aryan Chavan

2. Mention the names of the Submitted files :

1 parser.c	2 parse_table.h	3 parse_table.c
4 lexer.c	5 hash_Table.c	6 testcase1.txt
7 linked_list.c	8 Stack.c	9 testcase2.txt
10 linked_list.h	11 driver.c	12 testcase3.txt
13 grammar.txt	14 lexer.h	15 testcase4.txt
16 parse_tree.c	17 Makefile	18 testcase5.txt
19 testcase8.txt	20 testcase13.txt	21 testcase6.txt
22 testcase9.txt	23 testcase12.txt	24 testcase7.txt
25 testcase10.txt	26 testcase11.txt	27 coding details.pdf

3. Total number of submitted files: 27 (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: We created a DFA by taking each char from buffer

[B]. DFA implementation (State transition using switch case, graph, transition table, any other (specify): Switch Case

[C]. Keyword Handling Technique: Every alpha-numeric token is searched in lookup table to check if it is a keyword or not.

[D]. Hash function description, if used for keyword handling: Polynomial Rolling Hash Function, we take each character of string multiply with a multiplier. The multiplier keeps on increasing in power. The total sum is used as hash code.

[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 Lexeme of length >20, characters not present in language, half operators and half nums.

[H].Data Structure Description for token Info (in maximum two lines):

Name of token (Enum) and union of {char* id , int num and double rnum}

[I]. Interface with parser After parsing each token the lexer is called which tokenizes the next token and ignores lexical error token

7. Parser Details:

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

i. grammar : custom Struct named Symbol is created which contains a linked list containing the first and follow and details regarding whether it is a terminal or not. A right pointer pointing to next symbol present in grammar rule

ii. parse table 2d array of pointers of a custom struct called struct (described above)

iii. parse tree: (Describe the node structure also) Tree implemented using linked list. Each node stores the current symbol, its parent and next sibling.

iv. Parsing Stack node structure : Stack using linked list. Linked list is made of node struct. Node contains pointer to a symbol, the next node

v. Any other (specify and describe)

[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)

The tree is printed in INORDER, i.e., left child -> parent -> other siblings

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

i. Data structure for original grammar rules Array of pointers to symbols

ii. FIRST and FOLLOW sets computation automated (yes /no) Yes

iii. Data structure for representing sets Linked List

iv. Time complexity of computing FIRST sets Exponential

v. Name the functions (if automated) for computation of First and Follow sets

compute_first() and compute_follow()

vi. If computed First and Follow sets manually and represented in file/function (name that) _NA .

[D]. Error Handling

i. Attempted (yes/ no):____Yes_____

ii. Printing errors (All errors/ one at a time) :__All error_____

iii. Describe the types of errors handled

_1. Lexical Errors – Printing the line and exact token causing the error

2. Stack Top Terminal and Current Token doesn't match – stack is pooped

3.Stack Top Non-Terminal and Current Token doesn't have a rule in grammar – Synch Set is used

iv. Synchronizing tokens for error recovery (describe) There are 4 strategies adopted:

1. Using First Set as Synchronization Set

2. Using Follow Set as Synchronization Set

3. Using both First and Follow as Synchronization Set

4. Ignore tokens till ";" is found and pop stock until "Statement" or "Statements" is stack top with FIRST(Statement) or FIRST(Statements) as Sync set

v. Total number of errors detected in the given testcase t6(with_syntax_errors).txt

_____Twelve (12)_____

8. Compilation Details:

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

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

[C]. Mention the .c files that do not compile:_____NA_____

[D]. Any specific function that does not compile:_____NA_____

[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):_____Compiled successfully and running_____

_____

[B]. Execution time taken for

- t1.txt (in ticks) __12829_____ and (in seconds) ___0.013_____
- t2.txt (in ticks) __10438_____ and (in seconds) ___0.01_____
- t3.txt (in ticks) __11654_____ and (in seconds) ___0.012_____
- t4.txt (in ticks) __19750_____ and (in seconds) ___0.02_____
- t5.txt (in ticks) __29599_____ and (in seconds) ___0.03_____
- t6.txt (in ticks) __22440_____ and (in seconds) ___0.022_____

[C]. Gives segmentation fault with any of the test cases (1-6) uploaded on the course page. If yes, specify the testcase file name:_____NA_____

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

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

13. Declaration: We, _Aditya Sheth, Aryan Chavan, Kathan Patel, Mohit Makwana and Samay Gandhi_ (your names), declare that we have put our genuine efforts into 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 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 name below]

ID__2020A7PS0048P_____ Name: Mohit Makwana_____
ID__2020A7PS0058P_____ Name: _Kathan Patel_____
ID__2020A7PS0299P_____ Name: _Samay Gandhi_____
ID__2020A7PS1511P_____ Name: Aditya Sheth_____
ID__2020A7PS1692P_____ Name: Aryan Chavan_____

Date: __2nd of March, 2023_____

Should not exceed 4 pages.