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

Group No.
1

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: 2020A7PS0035P	Name: Shreekar Puranik
ID: 2020A7PS1205P	Name: Nikhil Pradhan
ID: 2020A7PS0146P	Name: Toshit Jain
ID: 2020A7PS0116P	Name: Ansh Gupta
ID: 2020A7PS1209P	Name: Sriram Ramanathan

2. Mention the names of the Submitted files :

1 Driver.c	7 Lexer.c	13 testcase1.txt
2 Parserdef.h	8 grammar.txt	14 testcase2.txt
3 Parser.h	9 MakeFile	15 testcase3.txt
4 Parser.c	10 README.txt	16 testcase4.txt
5 Lexerdef.h	11 Performa.pdf	17 testcase5.txt
6 Lexer.h	12 hash.h	18 testcase6.txt

3. Total number of submitted files: 18
4. Have you mentioned your names and IDs at the top of each file (and commented well)? ___Yes___ .
5. Have you compressed the folder as specified in the submission guidelines? Yes

6. Lexer Details:

[A]. Technique used for pattern matching: We used a DFA to find the possible patterns (Lexemes)

[B]. DFA implementation : State transition using switch case

[C]. Keyword Handling Technique: Used Hashmap mapping keys (keywords) to enum type.

[D]. Hash function description, if used for keyword handling: hash indexing using a function that gives sum of characters MOD 47. Collisions handled using linear probing.

[E]. Have you used twin buffers ? Yes

[F]. Lexical error handling and reporting : Yes

[G]. Describe the lexical errors handled by you:

i) Identifier is too long

ii) identifier.num

iii) num.abc

iv) num.numE(+/-)

v) single '='

vi) single '!'

vii) single '.'

viii) Symbol not present in Alphabet

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

struct Token contains :- line No.(int) , token type (enum), and a union data structure containing three possible data types - NUM(int) , RNUM(double) and string(for everything else)

[I]. Interface with parser: parser calls function getNextToken() , which returns a pointer to the next token, if the pointer has reached EOF it returns NULL.

7. Parser Details:

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

i. grammar : Grammar is structured in the form of an array of LinkedLists. Each grammar rule is given a separate index. The head points to the LHS non-terminal, then subsequently, each node points to the RHS grammar terminals/non-terminals in order from left to right.

ii. parse table : The parse table is an array of size (no of non-terminals) *(no of terminals - 1) since epsilon won't have a column in the parse table. It is a 2-D array of integers, each element in the table is either the rule number, or -1 if there is no rule corresponding to that element in the table.

iii. parse tree: (Describe the node structure also): Parse Tree is made up of TreeNodes. Each TreeNode data structure has 5 data points - the type of node (i.e. Terminal, Non-Terminal), the rule no. corresponding to the rules which it derives, pointer to the child node, pointer to the sibling node, and a union data structure consisting of either the non-terminal(enum), or the terminal(struct Token).

iv. Parsing Stack node structure : We have a pointer to the next stack node, and we also store a pointer to the corresponding TreeNode in the stack node (i.e. it is a stack of pointers to treenodes)

v. Any other (specify and describe): The LinkedList is made up of ListNode data structure. Each ListNode contains the grammarchar stored in the node, as well as a pointer to the next node. the grammarchar is a struct storing the type (Terminal or Non-Terminal) and a union data structure storing either the non-terminal or the tokentype

[B].Parse tree

i. Constructed : Yes

ii. Printing as per the given format : Yes

iii. Describe the order you have adopted for printing the parse tree nodes (in maximum two lines): In-order traversal such that we first print the leftmost child, then we print the details of the current node, then printing the details of the rest of the children subtrees after the leftmost one

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

i. Data structure for original grammar rules: Array of LinkedLists, where each rule gets its own index
ii. FIRST and FOLLOW sets computation automated : Yes
iii. Data structure for representing sets: LinkedLists
iv. Time complexity of computing FIRST sets: O(m*n*k) where
    1. m = No. of rules in the Grammar
    2. n = No. of Non-Terminals (No of first sets)
    3. k = Maximum length of a production rule
v. Name the functions (if automated) for computation of First and Follow sets: computefirstandfollow() calls computeFirst() and computeFollow() for each non-terminal
vi. If computed First and Follow sets manually and represented in file/function (name that): Not applicable.

[D]. Error Handling

i. Attempted : Yes
ii. Printing errors : Printing all the errors at a time
iii. Describe the types of errors handled :
    1. Lexical errors
    2. Stack empty but tokens still to be read
    3. Reached end of file but stack not empty
    4. Token doesn't match with top of stack terminal
    5. Rule not found in parse table.
iv. Synchronizing tokens for error recovery (describe): Follow sets corresponding to each non-terminal.
v. Total number of errors detected in the given testcase t6(with_syntax_errors).txt: 12

8. Compilation Details:

[A]. Makefile works : Yes

[B]. Code Compiles : Yes

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

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

[E]. Ensured the compatibility of your code with the specified gcc version(yes/no): No. We have checked with GCC version 9.4.0 on Ubuntu 20.04.1

9. Driver Details: Does it take care of the options specified earlier : Yes

10. Execution

[A]. Status (describe in maximum 2 lines): Working

[B]. Execution time taken for

- testcase1.txt (in ticks) : 2657 and (in seconds) : 0.002657
- testcase2.txt (in ticks) : 1565 and (in seconds) : 0.001565
- testcase3.txt (in ticks) : 1815 and (in seconds) : 0.001815
- testcase4.txt (in ticks) : 2380 and (in seconds) : 0.00238
- testcase5.txt (in ticks) : 2348 and (in seconds) : 0.002348
- testcase6.txt (in ticks) : 2631 and (in seconds) : 0.002631

[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 Faults

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

12. Are you availing the lifeline : Yes

13. Declaration: We, Shreekar Puranik, Nikhil Pradhan, Toshit Jain, Ansh Gupta, and Sriram Ramanathan, 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.

ID: 2020A7PS0035P Name: Shreekar Puranik

ID: 2020A7PS1205P Name: Nikhil Pradhan

ID: 2020A7PS0146P Name: Toshit Jain

ID: 2020A7PS0116P Name: Ansh Gupta

ID: 2020A7PS1209P Name: Sriram Ramanathan

Date: 03-03-2023

Should not exceed 4 pages