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

**DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS**

**Principles of Programming Languages (CS F301)**

**Group No.**

**12**

**I Semester 2020-21**

**Assignment-1 Code Submission**

**Coding Details**

**(October 29, 2020)**

1. IDs and Names of team members

ID: 2018A7PS0240P Name: Akshay Gundewar

ID: 2018A7PS0228P Name: Kanishk Patidar

ID: 2018A7PS0282P Name: Siddhant Jain

ID: 2018A7PS0339P Name: Utkarsh Srivastava

1. Mention the names of the Submitted files :

1 grammar.txt 7 parsetree1.c 13 t3.txt

2 grammar.h 8 traverseparsetree.h 14 t4.txt

3 grammar.c 9 traverseparsetree.c 15 t5.txt

4 tokenizer.h 10 coding details proforma.docx 16 t6.txt

5 tokenizer.c 11 t1.txt 17 driver.c

6 parsetree1.h 12 t2.txt

1. Total number of submitted files: \_\_\_\_17\_\_\_ (All files should be in **ONE folder** named exactly as Group\_#, # is your group number)
2. 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]
3. Have you compressed the folder as specified in the submission guidelines? (yes/no)\_\_\_\_\_yes\_\_\_\_\_\_
4. Have you ensured that the folder does not have any \*.o file or any executable file? (yes/no)\_\_\_yes\_\_\_\_
5. **Grammar and token stream**

Total number of production rules: \_\_\_\_\_\_59\_\_\_\_\_\_\_

Total number of nonterminals: \_\_\_\_\_\_\_\_30\_\_\_\_\_\_\_\_

Total number of terminals: \_\_\_\_\_\_\_\_31\_\_\_\_\_\_\_\_\_\_

Grammar.txt file created [yes/no]:\_\_\_\_\_\_\_\_yes\_\_\_\_\_\_\_\_

Nonterminal symbols enumerated [yes/no]:\_\_\_\_\_\_\_no\_\_\_\_\_\_\_

Terminal symbols enumerated [yes/no]:\_\_\_\_\_\_\_yes\_\_\_\_\_\_\_\_\_

Grammar data structure populated successfully [yes/no]:\_\_\_\_\_\_yes\_\_\_\_\_\_\_

Tokenstream created [yes/no]:\_\_\_\_\_\_\_\_yes\_\_\_\_\_\_\_\_\_\_

1. **Which functions have you implemented?**
   1. ***readGrammar ( ) [yes/no] \_\_\_\_\_\_\_\_yes\_\_\_\_\_\_\_\_\_\_\_***
   2. ***tokeniseSourcecode ( ) [yes/no] \_\_\_\_\_\_\_\_yes\_\_\_\_\_\_\_\_\_\_\_***
   3. ***createParseTree ( ) [yes/no] \_\_\_\_\_\_\_\_\_yes\_\_\_\_\_\_\_\_\_\_***
   4. ***traverseParseTree ( ) [yes/no] \_\_\_\_\_\_\_yes\_\_\_\_\_\_\_\_\_***
   5. ***printParseTree ( ) [yes/no] \_\_\_\_\_\_\_\_yes\_\_\_\_\_\_\_\_\_***
   6. ***printTypeExpressionTable ( ) [yes/no] \_\_\_\_\_\_\_yes\_\_\_\_\_\_\_\_\_***
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)

We did a Pre Order traversal on the tree. Each line describes symbol name, bool is\_terminal, depth of the node as first three parameters. If it is a leaf node we print the name of the lexeme and the line number, else we print type expression.

1. **Type Expression Table**

[A]. Constructed (yes/no):\_\_\_yes\_\_\_\_

[B]. Implemented as (lookup table/ hash table):\_\_\_\_\_\_\_\_\_lookup table\_\_\_\_\_\_\_\_\_\_

[C]. Printing as per the given format (yes/no): \_\_\_\_\_\_\_yes\_\_\_\_\_\_\_

[C]. Describe the structure of the type expression accommodating all types (in maximum two lines)

→Type expression statement is used with only type field for primitive types, along with proper dimension ranges and element type for rectangular arrays, and also including proper row/depth sizes for jagged arrays.

1. **Compilation Details:**
   1. Implemented in multiple files / single file:\_\_\_\_\_\_\_multiple files\_\_\_\_\_\_\_\_
   2. Makefile works (yes/no):\_\_\_\_no\_\_\_\_
   3. Code Compiles (yes/ no):\_\_\_\_\_yes\_\_\_\_\_\_
   4. Mention the .c files that do not compile:\_\_\_\_\_\_\_\_\_\_none\_\_\_\_\_\_\_\_\_\_\_\_
   5. Any specific function that does not compile:\_\_\_\_\_\_\_\_\_none\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   6. Ensured the compatibility of your code with the specified gcc version(yes/no)\_\_\_yes\_\_\_\_
   7. Give below the exact commands to compile your code :

Type the following commands in the terminal for execution of a testcase named “filename.txt”:-

gcc driver.c tokenizer.c parsetree1.c grammar.c traverseparsetree.c

./a.out filename.txt

1. **Driver Details**: Does it take care of the options specified earlier(yes/no):\_\_\_\_yes\_\_\_\_\_
2. **Execution** 
   1. Status (describe in maximum 2 lines): Executing successfully.
   2. Gives segmentation fault with any of the test cases (1-6) uploaded on the course page. If yes, specify the testcase file name:\_\_\_no\_\_\_
   3. Command line arguments used for input file (yes/no):\_\_\_\_\_yes\_\_\_\_\_\_\_
3. Specify the language features your code is not able to handle (in maximum one line):

Not able to get type checking errors in assignment statements.

1. Are you availing the lifeline (Yes/No): \_\_\_\_\_Yes\_\_\_\_\_
2. Declaration: We, Akshay Gundewar, Kanishk Patidar, Siddhant Jain, and Utkarsh Srivastava declare that we have put our genuine efforts in creating the 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: 2018A7PS0240P Name: Akshay Gundewar

ID: 2018A7PS0228P Name: Kanishk Patidar

ID: 2018A7PS0282P Name: Siddhant Jain

ID: 2018A7PS0339P Name: Utkarsh Srivastava

Date: \_\_\_29-Oct-2020\_\_\_

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

Should not exceed 3 pages.