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

Group No.

20

Compiler Construction (CS F363)
II Semester 2018-19
Compiler Project (Stage-2 Submission)
Coding Details
(April 15, 2019 using LifeLine)

Instruction: Write the details precisely and neatly. Places where you do not have anything to mention, please write NA for Not Applicable.

1. IDs and Names of team members

ID: 2016A7PS0140P Name: AMIT BANSAL

D: 2016A7PS0031P Name: VEDANT PATWARY

ID: 2016A7PS0004P Name: ABHILASH NEOG

ID: 2016A7PS0112P Name: ABHIMANYU SINGH SHEKHAWAT

2. Mention the names of the Submitted files (Include Stage-1 and Stage-2 both)

1. lexerDef.h
2. lexer.h
3. lexer.c
4. HashTable.h
5. HashTable.c
6. parserDef.h
7. parser.h
8. parser.c
9. DFA.jpg
10. stack.h
11. stack.c
12. makefile
13. SemanticRules.pdf
14. ast.h
15. ast.c
16. semanticAnalyser.h
17. symbolTableDef.h
18. symbolTable.h
19. symbolTable.c
20. typeCheckerDef.h
21. driver.c
22. typeChecker.c
23. typeChecker.h
24. semanticAnalyser.c
25. testcase1.txt
26. testcase2.txt
27. testcase3.txt
28. grammar.txt
29. main1.txt
30. main2.txt
31. main3.txt
32. main3.txt

3. Total number of submitted files: _____32______ (All files should be in ONE folder named exactly as Group_#, # is your group number)

4. Have you compressed the folder as specified in the submission guidelines? (yes/no) ______yes__________

5. Status of Code development: Mention 'Yes' if you have developed the code for the given module, else mention 'No'.

   a. Lexer (Yes/No): __________YES_________________
   b. Parser (Yes/No): ___________YES________________
   c. Abstract Syntax tree (Yes/No): ____YES___________
   d. Symbol Table (Yes/ No): ________YES____________
   e. Type checking Module (Yes/No): _______YES____________
   f. Semantic Analysis Module (Yes/ no): _____YES_______ (reached LEVEL = 4 as per the details uploaded)
   g. Code Generator (Yes/No): ______NO___________

6. Execution Status:

   a. Code generator produces code.asm (Yes/ No): ____________No_____________
   b. code.asm produces correct output using NASM for testcases (Main#.txt, #:1-4): ____No________
   c. Semantic Analyzer produces semantic errors appropriately (Yes/No): _____Yes_______
   d. Type Checker reports type mismatch errors appropriately (Yes/ No): ______Yes________
   e. Symbol Table is constructed (yes/no) ___Yes_____ and printed appropriately (Yes /No): ______Yes__________
   f. AST is constructed (yes/ no) ______Yes_____ and printed (yes/no) ____Yes______
   g. Name the test cases out of 7 as uploaded on the course website for which you get the segmentation fault (testcase#.txt ; # 1-3 and Main@.txt ; @:1-4):_______________N/A___________________________

7. Data Structures (Describe in maximum 2 lines and avoid giving C definition of it)
    a. AST node structure: Similar to parse tree node with field for rule number, terminal and non-terminal. Separate pointers to distinguish sibling, child, addr, syn and inh node and an identifier pointer to symbol table entry
    b. Symbol Table structure: 3 separate table for global variable, record and functions with fields for offset, width, name, pointer to local variable table in a given scope in function table etc. function table also has separate field for number of input and output parameters
    c. Data structure for global variables: Separate hash table is implemented for global variable via traversing AST.
    d. Record type expression structure: Separate hash table with instance of every record is stored with entry. Individual type index is maintained to assign a type to record for typechecking
    e. Input parameters type structure: Instance of number of input parameters for each function is maintained.
    f. Output parameters type structure: Instance of number of output parameters for each function is maintained
    g. Structure for maintaining the three address code (if created): __________N/A______________________
    h. Any other interesting data structure used: N/A

    ____

8. Semantic Checks: Mention your scheme NEATLY for testing the following major checks
    a. Variable not Declared: Check in local, global and record symbol table if absent report error
    b. Multiple declarations: Check in symbol tables if already declared then report error
    c. Number and type of input and output parameters:Check function table entry while traversing AST and match
    d. assignment of value to the output parameter in a function: A binary array is maintained for output parameter, bit for index if it is initiated and reset if not.
    e. function call semantics: Node for Calling function should be right of called function in AST
    f. type checking: Iterate over assignment statements, arithmetic expressions and singleOrRecID subtree while checking for individual operator and get type from symbol table. Each new record will have new type. This is achieved by keeping indexType variable in record table entry.
    g. return semantics: Check function table while traversing AST
    h. Recursion : By comparing lexemes i.e. name of function with calling and called function if both are same than report error
    i. function overloading: While populating symbol table if function is already present in table report error.
    j. 'while' loop semantics: Maintain a list of loop variable check for change if no list changes report error.
    k. record data type semantics and type checking of record type variables: Traverse AST assign new type and check via traversing singleOrRecId subtree.
    l. register allocation: ______________________N/A____
    .m. Scope of variables and their visibility: List of variable is maintained for each function entry in function table.
9. Compilation Details:
    a. Makefile works (yes/No): ______Yes_____
    b. Code Compiles (Yes/ No): ________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) _______Yes_____

10. Driver Details: Does it take care of the options specified earlier?(yes/no):_____Yes______
11. Specify the language features your compiler is not able to handle (in maximum one line)
    _All the language features are being handled by the compiler
12. Are you availing the lifeline (Yes/No): ______ YES
13. Write exact command you expect to be used for executing the code.asm using NASM simulator: N/A
    ____

14. Strength of your code(Tick the boxes where applicable):
(a) correctness (b) completeness (c) robust (d) Well documented (e) readable (f) strong data structure (f) Good programming style (indentation, avoidance of goto stmts etc) (g) modular (h)space and time efficient
________________All of the above_________________

15. Any other point you wish to mention: _____________________________N/A_______________________________
____
____

16. Declaration: We, AMIT BANSAL, ABHIMANYU SINGH SHEKHAWAT, VEDANT PATWARY and ABHILASH NEOG declare that we have put our genuine efforts in creating the compiler project code and have submitted the code developed by us. 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.

Date: 15 April 2019(using LifeLine)
(Not to exceed beyond 3 pages)