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

**Compiler Construction (CS F363)**
**II Semester 2022-23**
**Compiler Project (Stage-2 Submission)**
**Coding Details**
**(April 12, 2023)**

**Group number ______________50_______________(Write your group number here)**

*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:____ 2019B3A70489P_________ Name:________ Harsh Pandey ________________________________

ID:_____ 2019B3A70470P _________ Name:_________ Aryan Puwar _______________________________

ID:______ 2019B3A70562P __________ Name:_________ Tejas Deshpande ___________________________

ID:______ 2019B2A70898P ________ Name:_________ Harshit Gupta _____________________________

ID:_______ 2019B3A70819P ________ Name:_________ Krish Vora ________________________________

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

1 _grammar.txt
2______lexer.c_________
3____driver.c_________
4__makefile________
5__parser.c_____________
6__parser.h_________
7___structs.h______
8_lexer.h________
9___ast.c___
10_dfa.pdf______________
11__firstfollow.pdf________
12_cocoastgrp50.pdf______
13____t1.txt__________
14_____t2.txt_________
15__t3.txt__________
16________t4.txt______
17___t5.txt________
18___t6.txt_________
19_____t7.txt_________
20____t8.txt__________
21________t9.txt____
22______t10.txt_______
23_______c1.txt_______
24______c2.txt_________
25 c3.txt 26 c4.txt 27 c5.txt 28 c6.txt 29 c7.txt 30 c8.txt 31 c9.txt 32 c10.txt 33 c11.txt

3. Total number of submitted files: ____33____ (All files should be in **ONE** folder named exactly as Group number)
4. Have you mentioned names and IDs of all team members 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. **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):____No___________
   f. Semantic Analysis Module (Yes/ no):_____No _____(reached LEVEL _NA_ as per the details uploaded)
   g. Code Generator (Yes/No):_____No_______

7. **Execution Status**:
   a. Code generator produces code.asm (Yes/ No):NA
   b. code.asm produces correct output using NASM for testcases (C#.txt, #:1-11): NA

c. Semantic Analyzer produces semantic errors appropriately (Yes/No):____NA______

d. Static Type Checker reports type mismatch errors appropriately (Yes/ No):___Not properly functioning______

e. Dynamic type checking works for arrays and reports errors on executing code.asm (yes/no): __No________________

f. Symbol Table is constructed (yes/no)_Yes____and printed appropriately (Yes /No):___Yes________

g. AST is constructed (yes/ no) __Yes______and printed (yes/no) __Yes____

h. Name the test cases out of 21 as uploaded on the course website for which you get the segmentation fault (t#.txt ; # 1-10 and c@.txt ; @:1-11):__________________________________________

8. **Data Structures** (Describe in maximum 2 lines and avoid giving C definition of it)
   a. AST node structure So we are basically storing the value of each node and an attribute name as syn list and syn address. So basically some ast nodes have syn list and others have address.

   b. Symbol Table structure: It is basically similar to linked list of a structure that has all the attributes that are required for a symbol table row.

   c. array type expression structure: It is incorporated in the symbol table only. We have used a bool attribute as isArray is that is true than it will have all the fields required for the array type.

   d. Input parameters type structure: It is also incorporated in the symbol table data structure only.

   e. Output parameters type structure: It is also incorporated in the symbol table data structure only.

   f. Structure for maintaining the three address code(if created) :________________________________

   ______________________________________________________________________________________

9. **Semantic Checks:** Mention your scheme NEATLY for testing the following major checks (in not more than 5-10 words)[ Hint: You can use simple phrases such as 'symbol table entry empty', 'symbol table entry already found populated', 'traversal of linked list of parameters and respective types' etc.]
   a. Variable not Declared :symbol table entry empty

   b. Multiple declarations: symbol table entry already found populated

   c. Number and type of input and output parameters:symbol table entry already found populated

   d. assignment of value to the output parameter in a function symbol table entry already found populated

   e. function call semantics:symbol table entry already found populated

   f. static type checking :

   g. return semantics:__________________________________________________________________________

   h. Recursion :________________________________________________________________________________

   i. module overloading:_________________________________________________________________________

   j. 'switch' semantics :________________________________________________________________________

   k. 'for' and 'while' loop semantics: __________________________________________________________

   l. handling offsets for nested scopes:_________________________________________________________

   m. handling offsets for formal parameters:_____________________________________________________

   n. handling shadowing due to a local variable declaration over input parameters:___________________

   ______________________________________________________________________________________

   o. array semantics and type checking of array type variables: _________________________________

________________________________________________________________________________

p. Scope of variables and their visibility :____________________________________________________

q. computation of nesting depth:____________________________________________________

10. Code Generation:

a. NASM version as specified earlier used (Yes/no):____No________________
b. Used 32-bit or 64-bit representation:____________NA_________________________
c. For your implementation: 1 memory word = ______NA__________(in bytes)
d. Mention the names of major registers used by your code generator:
    - For base address of an activation record: __________________
    - for stack pointer:____________________________________
    - others (specify):____________________________________
e. Mention the physical sizes of the integer, real and boolean data as used in your code generation module
size(integer): ______________________(in words/ locations), ______________(in bytes)
size(real): _________________________(in words/ locations), ______________(in bytes)
size(booelan): ______________________(in words/ locations), ______________(in bytes)

f. How did you implement functions calls?(write 3-5 lines describing your model of implementation)
________________________________________________________________________________
________________________________________________________________________________
________________________________________________________________________________

g. Specify the following:
    - Caller's responsibilities:________________________________________________
    - Callee's responsibilities:________________________________________________

h. How did you maintain return addresses? (write 3-5 lines): __________________________________
________________________________________________________________________________
________________________________________________________________________________
________________________________________________________________________________

i. How have you maintained parameter passing? How were the statically computed offsets of the parameters used by the callee? ____________________________________________________

j. How is a dynamic array parameter receiving its ranges from the caller? ____________________

k. What have you included in the activation record size computation? (local variables, parameters, both):
________________________________________________________________________________

l. register allocation (your manually selected heuristic) :______________________________________
________________________________________________________________________________

m. Which primitive data types have you handled in your code generation module?(Integer, real and boolean):__________________________________________________________________

n. Where are you placing the temporaries in the activation record of a function? __________________
________________________________________________________________

11. **Compilation Details**:

a. Makefile works (yes/No):__Yes______

b. Code Compiles (Yes/ No):__Yes________

c. Mention the .c files that do not compile:_____________________________________________

d. Any specific function that does not compile:________________________________________

e. Ensured the compatibility of your code with the specified versions [GCC, UBUNTU, NASM] (yes/no)_____Yes__

12. Execution time for compiling the test cases [lexical, syntax and semantic analyses including symbol table creation, type checking and code generation] :

    i. t1.txt (in ticks) 2086 and (in seconds) 0.002086

    ii. t2.txt (in ticks) 3231 and (in seconds) 0.003231

    iii. t3.txt (in ticks) 7674 and (in seconds) 0.007674

    iv. t4.txt (in ticks) 6656 and (in seconds) 0.006656

    v. t5.txt (in ticks) 8798 and (in seconds) 0.008798

    vi. t6.txt (in ticks) and (in seconds)

    vii. t7.txt (in ticks) 4332 and (in seconds) 0.004332

    viii. t8.txt (in ticks) 8763 and (in seconds) 0.008763

    ix. t9.txt (in ticks) and (in seconds)

    x. t10.txt (in ticks) 4387 and (in seconds) 0.004387

13. **Driver Details**: Does it take care of the **TEN** options specified earlier?(yes/no):__Not all_____
14. Specify the language features your compiler is not able to handle (in maximum one line)
    Code generation part is not handled_______________________________________________
15. Are you availing the lifeline (Yes/No): ____No__________
16. Write exact command you expect to be used for executing the code.asm using NASM simulator [We will use these directly while evaluating your NASM created code]
    NA
    ______________________________________________________________________________
17. **Strength of your code**(Strike off where not applicable): (a) correctness (b) completeness (c) robustness (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
18. Any other point you wish to mention: ____________________________________________________

    ______________________________________________________________________________

    ______________________________________________________________________________

19. Declaration: We, Harsh Pandey, Aryan Puwar, Tejas Deshpande, Harshit Gupta, Krish Vora 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_______2019B3A70489P________ Name:____Harsh Pandey _________
    ID_______2019B3A70470P __________ Name:____Aryan Puwar __________
    ID_______2019B3A70562P ___________ Name:_____Tejas Deshpande ______
    ID_______2019B2A70898P ___________ Name:_____Harshit Gupta ________

ID________2019B3A70819P ___________ Name:_______Krish Vora ___________

Date: _____12/03/2023___ Group number_____50______

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

Should not exceed 6 pages.