# Programming Assignment #4

## Deadline: 23rd April March, 2024

## Full Marks: 30

1. <prog> ::= PROGRAM <prog-name> VAR <dec-list> BEGIN <stmt-list> END.
2. <prog-name> ::= **id**
3. <dec-list> ::= <dec> | <dec-list> ; <dec>
4. <dec> ::= <id-list> : <type>
5. <type> ::= INTEGER | REAL
6. <id-list> ::= **id** | <id-list>,**id**
7. <stmt-list> ::= <stmt> | <stmt-list>; <stmt>
8. <stmt> ::= <assign> | <read> | <write> | <for>
9. <assign> ::= **id** := <exp>
10. <exp> ::= <term> | <exp> + <term> | <exp> - <term>
11. <term> ::= <factor> | <term>*<factor> | <term> DIV <factor>
12. <factor> ::= **id** | (<exp>)
13. <read> ::= READ(<id-list>)
14. <write> ::= WRITE(<id-list>)
15. <for> ::= FOR <index-exp> DO <body>
16. <index-exp> ::= **id** := <exp> TO <exp>
17. <body> ::= <stmt> | BEGIN <stmt-list> END

We are being provided above a simplified PASCAL grammar in BNF (Backus-Naur Form).

Generate the parser of the above grammar using YACC and LEX while handling all syntax and semantic errors augmenting the grammar with proper attributes.

Use the PASCAL program given in the earlier assignment as test case incorporating syntax and semantic errors.

Token coding scheme for the above grammar is tabulated as below:

| **TOKEN** | **CODE** |
|---|---|
| PROGRAM | 1 |
| VAR | 2 |
| BEGIN | 3 |
| END | 4 |
| END. | 5 |
| INTEGER | 6 |
| REAL | 7 |
| FOR | 8 |
| READ | 9 |
| WRITE | 10 |
| TO | 11 |
| DO | 12 |
| ; | 13 |
| : | 14 |
| , | 15 |
| := | 16 |
| + | 17 |
| - | 18 |
| * | 19 |
| DIV | 20 |
| ( | 21 |
| ) | 22 |
| **id** | 23 |

==========