

LL(1) Modified Tiger Grammar

- (1) `<tiger-program> -> let <declaration-segment> in <stat-seq> end`
- (2) `<declaration-segment> -> <type-declaration-list> <var-declaration-list> <funct-declaration-list>`
- (3) `<type-declaration-list> -> NULL`
- (4) `<type-declaration-list> -> <type-declaration> <type-declaration-list>`
- (5) `<var-declaration-list> -> NULL`
- (6) `<var-declaration-list> -> <var-declaration> <var-declaration-list>`
- (7) `<funct-declaration-list> -> NULL`
- (8) `<funct-declaration-list> -> <funct-declaration> <funct-declaration-list>`
- (9) `<type-declaration> -> type id = <type> ;`
- (10) `<type> -> <type-id>`
- (11) `<type> -> array [INTLIT] of <type-id>`
- (12) `<type-id> -> int`
- (13) `<type-id> -> string`
- (14) `<type-id> -> id`
- (15) `<var-declaration> -> var <id-list> : <type-id> <optional-init> ;`
- (16) `<id-list> -> id <id-list-tail>`
- (17) `<id-list-tail> -> , id <id-list-tail>`
- (18) `<id-list-tail> -> NULL`
- (19) `<optional-init> -> NULL`
- (20) `<optional-init> -> := <const>`
- (21) `<funct-declaration> -> function id (<param-list>) <ret-type> begin <stat-seq> end ;`
- (22) `<param-list> -> NULL`
- (23) `<param-list> -> <param> <param-list-tail>`
- (24) `<param-list-tail> -> NULL`
- (25) `<param-list-tail> -> , <param> <param-list-tail>`
- (26) `<ret-type> -> NULL`
- (27) `<ret-type> -> : <type-id>`
- (28) `<param> -> id : <type-id>`

- (30) <stat-seq> -> <stat> <stat-seq>
- (31) <stat-seq> -> NULL

- (32) <stat> -> id <stat-id> ;
- (33) <stat-id> -> (<expr-list>)
- (34) <stat-id> -> <lvalue> := <stat-id-tail>
- (81) <stat-id-tail> -> id <stat-id-tail-tail>
- (82) <stat-id-tail> -> <expr>
- (83) <stat-id-tail-tail> -> (<expr-list>)
- (84) <stat-id-tail-tail> -> <lvalue>
- (35) <stat> -> if <expr> then <stat-seq> <stat-tail>
- (36) <stat-tail> -> endif ;
- (37) <stat-tail> -> else <stat-seq> endif ;
- (38) <stat> -> while <expr> do <stat-seq> enddo ;
- (39) <stat> -> for id := <expr> to <expr> do <stat-seq> enddo ;
- (40) <stat> -> break ;

- (41) <expr> -> -<expr>
- (42) <expr> -> <andorterm>

- (43) <andorterm> -> <ineqterm> <andorterm2>
- (44) <andorterm2> -> <andorop> <ineqterm> <andorterm2>
- (45) <andorterm2> -> NULL
- (46) <andorop> -> |
- (47) <andorop> -> &

- (48) <ineqterm> -> <addterm> <ineqterm2>
- (49) <ineqterm2> -> <ineq> <addterm> <ineqterm2>
- (50) <ineqterm2> -> NULL
- (51) <ineq> -> <>
- (52) <ineq> -> =
- (53) <ineq> -> <
- (54) <ineq> -> >
- (55) <ineq> -> <=
- (78) <ineq> -> >=
- (79) <ineq> -> !=

- (56) <addterm> -> <multterm> <addterm2>
- (57) <addterm2> -> <addopp> <multterm> <addterm2>
- (58) <addterm2> -> NULL
- (59) <addopp> -> +
- (60) <addopp> -> -

- (61) <multterm> -> <factor> <multterm2>
- (62) <multterm2> -> <multop> <factor> <multterm2>
- (63) <multterm2> -> NULL

(64) <multop> -> /

(65) <multop> -> *

(66) <factor> -> (<expr>)

(67) <factor> -> <const>

(68) <factor> -> id <lvalue>

(69) <const> -> INTLIT

(70) <const> -> STRLIT

(71) <const> → nil

(72) <expr-list> -> NULL

(73) <expr-list> -> <expr> <expr-list-tail>

(74) <expr-list-tail> -> , <expr> <expr-list-tail>

(75) <expr-list-tail> -> NULL

(76) <lvalue> -> [<expr>] <lvalue>

(77) <lvalue> -> NULL