|  |  |  |  |
| --- | --- | --- | --- |
| Rule Number | Before | After | Method |
| 1 | PROG -> GLOBAL\_VARS FUNC\_PREDEFS FUNC\_FULL\_DEFS | PROG -> GLOBAL\_VARS FUNC\_PREDEFS FUNC\_FULL\_DEFS | None |
| 2 | GLOBAL\_VARS -> GLOBAL\_VARS VAR\_DEC | VAR\_DEC | GLOBAL\_VARS -> VAR\_DEC GLOBAL\_VARS’  GLOBAL\_VARS’ -> EPSILON | VAR\_DEC GLOBAL\_VARS’ | Left recursion |
| 3 | VAR\_DEC -> TYPE id; | TYPE id[DIM\_SIZES]; | VAR\_DEC -> TYPE id VAR\_DEC’  VAR\_DEC’ -> ; | [DIM\_SIZES]; | Left common prefixes |
| 4 | TYPE -> int | float | TYPE -> int | float | None |
| 5 | DIM\_SIZES ➔ int\_num I int\_num, DIM\_SIZES | DIM\_SIZES ➔ int\_num DIM\_SIZES’  DIM\_SIZES’ ➔ EPSILON | , DIM\_SIZES | Left common prefixes |
| 6 | FUNC\_PREDEFS ➔ FUNC\_PREDEFS FUNC\_PROTOTYPE; I FUNC\_PROTOTYPE; | FUNC\_PREDEFS ➔ FUNC\_PROTOTYPE ; FUNC\_PREDEFS’  FUNC\_PREDEFS’ ➔ EPSILON | FUNC\_PROTOTYPE ; FUNC\_PREDEFS’ | Left recursion |
| 7 | FUNC\_PROTOTYPE ➔ RETURNED\_TYPE id (PARAMS( | FUNC\_PROTOTYPE ➔ RETURNED\_TYPE id (PARAMS( | None |
| 8 | FUNC\_FULL\_DEFS ➔ FUNC\_WITH\_BODY FUNC\_FULL\_DEFS | FUNC\_WITH\_BODY | FUNC\_FULL\_DEFS ➔ FUNC\_WITH\_BODY FUNC\_FULL\_DEFS’  FUNC\_FULL\_DEFS’ ➔ EPSILON | FUNC\_FULL\_DEFS | Left common prefixes |
| 9 | FUNC\_WITH\_BODY ➔ FUNC\_PROTOTYPE COMP \_STMT | FUNC\_WITH\_BODY ➔ FUNC\_PROTOTYPE COMP\_STMT | None |
| 10 | RETURNED\_TYPE ➔ TYPE I void | RETURNED\_TYPE ➔ TYPE | void | None |
| 11 | PARAMS ➔ PARAM\_LIST I EPSILON | PARAMS ➔ PARAM\_LIST I EPSILON | None |
| 12 | PARAM\_LIST ➔ PARAM\_LIST, PARAM I PARAM | PARAM\_LIST ➔PARAM PARAM\_LIST’  PARAM\_LIST’ ➔ EPSILON | , PARAM PARAM\_LIST’ | Left recursion |
| 13 | PARAM ➔ TYPE id I TYPE id [ DIM\_SIZES] | PARAM ➔ TYPE id PARAM’  PARAM’ ➔ EPSILON | [DIM\_SIZES] | Left common prefixes |
| 14 | COMP \_STMT ➔ { VAR\_DEC\_LIST STMT\_LIST } | COMP \_STMT ➔ { VAR\_DEC\_LIST STMT\_LIST } | None |
| 15 | VAR\_DEC\_LIST ➔ VAR\_DEC\_LIST VAR\_DEC I EPSILON | VAR\_DEC\_LIST ➔ EPSILON | VAR\_DEC VAR\_DEC\_LIST | Left recursion |
| 16 | STMT\_LIST ➔ STMT\_LIST ; STMT I STMT | STMT\_LIST ➔ STMT STMT\_LIST’  STMT\_LIST’ ➔ EPSILON | ; STMT STMT\_LIST’ | Left recursion |
| 17 | STMT ➔VAR= EXPR I COMP \_STMT I IF \_STMT I CALL I RETURN\_STMT | Mid-Level:  STMT ➔ id(VAR)’ = EXPR | COMP \_STMT | IF \_STMT  | id ( ARGS ) | RETURN\_STMT  After:  STMT ➔ id STMT’ | COMP\_STMT | IF\_STMT |RETURN\_STMT  STMT’ ➔ VAR’ = EXPR | ( ARGS ) | Left common prefixes(Hidden) |
| 18 | IF\_STMT ➔ if (CONDITION) STMT | IF\_STMT ➔ if (CONDITION) STMT | None |
| 19 | CALL ➔ id ( ARGS ) | CALL ➔ id ( ARGS ) | None |
| 20 | ARGS ➔ ARG\_LIST I EPSILON | ARGS ➔ ARG\_LIST I EPSILON | None |
| 21 | ARG\_LIST ➔ ARG\_LIST , EXPR I EXPR | ARG\_LIST ➔ EXPR ARG\_LIST’  ARG\_LIST’ ➔ EPSILON | , EXPR ARG\_LIST’ | Left recursion |
| 22 | RETURN\_STMT ➔ return I return EXPR | RETURN\_STMT ➔ return RETURN\_STMT’  RETURN\_STMT’ ➔EPSILON | EXPR | Left common prefixes |
| 23 | VAR➔ id I id [ EXPR\_LIST] | VAR➔ id VAR’  VAR’ ➔ EPSILON | [ EXPR\_LIST] | Left common prefixes |
| 24 | EXPR\_LIST ➔ EXPR\_LIST , EXPR | EXPR | EXPR\_LIST ➔ EXPR EXPR\_LIST’  EXPR\_LIST’ ➔ EPSILON | , EXPR EXPR\_LIST’ | Left recursion |
| 25 | CONDITION ➔ EXPR rel\_op EXPR | CONDITION ➔ EXPR rel\_op EXPR | None |
| 26 | EXPR ➔ EXPR + TERM I TERM | EXPR ➔ TERM EXPR’  EXPR’ ➔ EPSILON | + TERM EXPR’ | Left recursion |
| 27 | TERM ➔ TERM \* FACTOR I FACTOR | TERM ➔ FACTOR TERM’  TERM’ ➔ EPSILON | \* FACTOR TERM’ | Left recursion |
| 28 | FACTOR➔ VAR I CALL I int\_ num I float\_num I (EXPR) | Mid-Level:  FACTOR ➔ id VAR’ | id (ARGS) | int\_num | float\_num | (EXPR)  After:  FACTOR ➔ id FACTOR’ | int\_num | float\_num | (EXPR)  FACTOR’ ➔ VAR’ | (ARGS) | Left common prefixes(Hidden) |