**Gramática para Sintáctico**

**=========================**

*(Lo tachado no iría porque esta repetido. Lo que está en verde es lo agrado o unificado. Falta continuar)*

**---------------------------------------------------------------------**

**Versión 3.0**

**Principal**

S’ -> S EOF

S -> GLOBALES SUBRUTINA PPAL SUBRUTINA

GLOBALES -> globales DECGL fin-globales; |

λ

SUBRUTINA -> FP |   
 λ

FP -> FUNCION |

PROCEDIMIENTO |

PPAL -> DECL BLOQUEP

**Funciones y procedimiento**

FUNCION -> ENCABEZADOF FUNCION’

ENCABEZADOF -> función PALABRA(PARAMS): TIPO;

FUNCION’ -> adelantado; FP |

DECL BLOQUEF

PROCEDIMIENTO -> procedimiento PROCEDIMIENTO’

PROCEDIMIENTO’ -> principal(); |

ENCABEZADOP PROCEDIMIENTO’’

PROCEDIMIENTO’’ -> adelantado; FP |

DECL BLOQUEP

ENCABEZADOP -> PALABRA(PARAMS);

**Bloques:**

BLOQUEF -> comenzar BLOQUE fin-funcion EXP; FP

BLOQUEP -> comenzar BLOQUE fin-proc; FP

BLOQUE -> LINEA BLOQUE |

λ

LINEA -> BLOQUESI |

BLOQUEM |

PALRES |

LLAMADOPROC |

ASIGN

BLOQUEM -> mientras EXPBLOQUE hacer BLOQUE

fin-mientras

BLOQUESI -> si EXPBLOQUE entonces BLOQUE BLOQUESI’

EXPBLOQUE -> EXPBOOL |

ESPAR |

ESIMPAR

BLOQUESI’ -> fin-si |

sino BLOQUE fin-si

PALRES -> LEER |

MOSTRAR |

MOSTRARLN

**Parametrización:**

PARAMS -> PARAMS’ |

λ

PARAMS’ -> PARAM PARAM’

PARAM -> TIPOPARAM PALABRA: TIPO

PARAM’ -> ,PARAM PARAMS’|

λ

TIPOPARAM -> TPARAM |

λ

TPARAM -> ref |

val

**Declaraciones:**

DECGL -> DECGLOBAL DECGL |

λ

DECGLOBAL -> VARSG |

CONSTS

DECL -> DECLARACIONES DECL |

λ

DECLARACIONES -> VARS |

CONSTS |

FP

**Asigancion**

ASIGN -> PALABRA ASIGAUX = EXP;

ASIGAUX -> : | /\* VARIABLE o VECTOR\*/

[EXP]:

**Constantes y variables**

CONSTS -> const CONSTS'

CONSTS' -> PALABRA: TIPO = NUMERO CONSTS'' |

PALABRA, CONSTS'

CONSTS'' -> , CONSTS' | ;

VARS -> var VARS'

VARS’ -> PALABRA TVAR

TVAR -> , VARS’ |

: TIPO INI VARS’’

VARS’’ -> , VARS’ |

;

VARSG -> var VARSG'

VARSG’ -> PALABRA TVARG

TVARG -> , VARSG’ |

: TIPO INI VARSG’’ |

[ NATURAL ] : TIPO;

VARSG’’ -> , VARSG’ |

;

INI -> = NUMERO |

λ

NUMERO -> ENTERO |

NATURAL

TIPO -> entero |natural

**Funciones requeridas**

AENTERO -> aentero(EXP)

ANATURAL -> anatural(EXP)

ESPAR -> par(EXP)

ESIMPAR -> impar(EXP)

MOSTRARLN -> mostrarln CADENA MOSTRARAUX’

MOSTRAR -> mostrar CADENA MOSTRARAUX’

MOSTRARAUX’ -> ,MSTRARAUX’’ |

;

MOSTRARAUX’’ -> CADENA MSTRARAUX’ |

PALABRA MOSTRARAUX’

LEER -> leer PALABRA;

**Terminales**

ENTERO -> /\* CONSUMO UN TOKEN DE TIPO ENTERO \*/

NATURAL -> /\* CONSUMO UN TOKEN DE TIPO NATURAL \*/

PALABRA -> /\* CONSUMO UN TOKEN DE TIPO PALABRA \*/

CADENA -> /\* CONSUMO UN TOKEN DE TIPO CADENA \*/

LLAMADOFUN -> /\* Consumir función si esta en tabla de

símbolossino error \*/

LLAMADOPROC -> /\* Consumir procedimiento si esta en tabla

de símbolos sino error \*/

**Expresiones:**

EXPBOOL -> EXP EXPBOOL' |

AND |

OR

EXPBOOL' -> = EXP |

< EXP |

!= EXP |

== EXP |

<< EXP |

!== EXP

AND -> and(EXPBOOL,EXPBOOL)

OR -> or(EXPBOOL,EXPBOOL)

EXP -> TERM EXP'

EXP' -> + TERM EXP' |

- TERM EXP' |

++ TERM EXP' |

-- TERM EXP' | λ

TERM -> FACT TERM'

TERM' -> \* FACT TERM' |

/ FACT TERM' |

\*\* FACT TERM' |

// FACT TERM'| λ

FACT -> (EXP) |

PALABRA VECT |

NUMERO |

LLAMADOFUN |

AENTERO |

ANATURAL

VECT -> [EXP] | λ

/\*

**RESTRICCION DEL LENGUAJE**:

El Llamado a funciones solo puede ser utilizado para asignaciones y no en condiciones de estructuras de control,

ej: SI dividirPor2(A) > 3 Entonces - NO ACEPTADO POR LA GRAMATICA

\*/