

TERMINALES	No TERMINALES
INSTRUCCIONES	func_exec
INSTRUCCION	def_entero
BLOQUE_SENTENCIAS	def_decimal
ASIGNACION	def_caracter
DECLARACION	def_cadena
MODIFICADOR	def_boolean
LLAMADA_FUNCION	esc_salto_linea
DECLARACION_METODOS	esc_comilla_doble
NO_IMPLEMENTADAS	esc_tabulacion
EXEC	esc_comilla_simple
SENTENCIA_CICLICA	esc_barra_invertida
SENTENCIA_CONTROL	incremento
EXPRESION	decremento
UNMENOS	suma
DATO	resta
TIPO_DATO	multiplicacion
IF	division
ELSE	potencia
ELIF	op_modulo
CONTROL_SWITCH	op_doble_igual
SWITCH	op_menor_igual
CASELIS	op_mayor_igual
CASES	op_diferencia
CASE	op_igual
DEFAULTS	op_menor
DEFAULT	op_mayor
SENTENCIA_CICLICA	dos_puntos
SENTENCIA_WHILE	pregunta_cierra
WHILE	op_or
SENTENCIA_FOR	op_and
ENCABEZADOFOR	op_not
FOR	par_abre
SENTENCIA_DOWHILE	par_cierra
DO	punto_coma
LLAMADA_FUNCION	llave_abre
LLAMADA_METODO	llave_cierra
INICIOMETODO	coma
PARAMETOS_LLAMADA	continue
FUNCION_PRINT	if
DECLARACION_METODOS	else
METODO	switch
VOID	case
PARAMETROS	break
NO_IMPLEMENTADAS	default
	while
	for

	do void print decimal entero cadena caracter verdadero falso identificador
--	---

Expresiones Regulares

```
[0-9]+("."[0-9]+)\b  
[0-9]+\b  
("\")(^[^"\\]|\\.)*("\")
```

Inicio de Gramática

```
INICIAL: INSTRUCCIONES EOF  
        | EOF;  
INSTRUCCIONES: INSTRUCCIONES INSTRUCCION {}  
               | INSTRUCCION {};
```

GRAMÁTICA

BLOQUE_SENTENCIAS: llave_abre INSTRUCCIONES llave_cierra

| llave_abre llave_cierra

|error llave_cierra

INSTRUCCION: ASIGNACION punto_coma

|DECLARACION punto_coma

|MODIFICADOR punto_coma

|LLAMADA_FUNCION punto_coma

|DECLARACION_METODOS

|NO_IMPLEMENTADAS

|EXEC punto_coma

|SENTENCIA_CICLICA

|SENTENCIA_CONTROL

DECLARACION: TIPO_DATO identificador

ASIGNACION: identificador op_igual EXPRESION

|TIPO_DATO identificador op_igual EXPRESION

EXPRESION: resta EXPRESION

|par_abre TIPO_DATO par_cierra EXPRESION

|EXPRESION suma EXPRESION

|EXPRESION resta EXPRESION

|EXPRESION multiplicacion EXPRESION

|EXPRESION division EXPRESION

|EXPRESION potencia EXPRESION

|EXPRESION op_modulo EXPRESION

|EXPRESION op_and EXPRESION

|EXPRESION op_or EXPRESION

|op_not EXPRESION

|EXPRESION op_mayor EXPRESION

| EXPRESION op_menor EXPRESION
| EXPRESION op_mayor_igual EXPRESION{\$\$
| EXPRESION op_menor_igual EXPRESION
| EXPRESION op_doble_igual EXPRESION
| EXPRESION op_diferencia EXPRESION
| EXPRESION pregunta_cierra EXPRESION dos_puntos EXPRESION
| EXPRESION incremento
| EXPRESION decremento
| DATO

DATO: decimal

| entero
| verdadero
| falso
| cadena
| caracter
| identificador

SENTENCIA_CONTROL: IF BLOQUE_SENTENCIAS else IF BLOQUE_SENTENCIAS ELIF

| IF BLOQUE_SENTENCIAS
| IF BLOQUE_SENTENCIAS ELSE BLOQUE_SENTENCIAS
| CONTROL_SWITCH

IF: if par_abre EXPRESION par_cierra

ELSE: else

ELIF: else IF BLOQUE_SENTENCIAS ELIF

| ELSE BLOQUE_SENTENCIAS

CONTROL_SWITCH: switch SWITCH llave_abre CASELIST llave_cierra

SWITCH: par_abre EXPRESION par_cierra

CASELIST: CASES CASELIST

| CASES

CASES: CASE INSTRUCCIONES

| CASE

| DEFAULTS;

CASE: case EXPRESION dos_puntos

DEFAULTS: DEFAULT INSTRUCCIONES

DEFAULT: default dos_puntos

SENTENCIA_CICLICA: SENTENCIA_WHILE

| SENTENCIA_FOR

| SENTENCIA_DOWHILE

SENTENCIA_WHILE: WHILE BLOQUE_SENTENCIAS

WHILE: while par_abre EXPRESION par_cierra

SENTENCIA_FOR: ENCABEZADOFOR BLOQUE_SENTENCIAS

ENCABEZADOFOR: FOR ASIGNACION punto_coma EXPRESION punto_coma MODIFICADOR
par_cierra

| FOR ASIGNACION punto_coma EXPRESION punto_coma ASIGNACION par_cierra

FOR: for par_abre

SENTENCIA_DOWHILE: DO BLOQUE_SENTENCIAS while par_abre EXPRESION par_cierra
punto_coma

DO:do

LLAMADA_FUNCION: FUNCION_PRINT

| LLAMADA_METODO

LLAMADA_METODO: INICIOMETODO PARAMETOS_LLAMADA par_cierra

| INICIOMETODO par_cierra

INICIOMETODO: identificador par_abre

PARAMETOS_LLAMADA: EXPRESION coma PARAMETOS_LLAMADA
| EXPRESION

FUNCION_PRINT: print par_abre EXPRESION par_cierra

DECLARACION_METODOS: METODO BLOQUE_SENTENCIAS

METODO: VOID identificador par_abre par_cierra

| VOID identificador par_abre PARAMETROS par_cierra

VOID: void

PARAMETROS: PARAMETRO coma PARAMETROS

| PARAMETRO;

PARAMETRO : TIPO_DATO identificador

EXEC: func_exec LLAMADA_METODO;

NO_IMPLEMENTADAS: continue;