

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA  
FACULTAD DE INGENIERÍA  
ORGANIZACIÓN DE LENGUAJES Y COMPILEDORES 1  
CATEDRÁTICO: ING. JOSE MOISES GRANADOS GUEVARA



PABLO DANIEL ALVARADO RODRIGUEZ

CARNÉ: 202130534

QUETZALTENANGO, FEBRERO 2026

## Análisis De Gramática para Lexer

Para el análisis de la gramática se tuvo en cuenta la entrada que debe de reconocer el intérprete y a partir de esta se derivó las definiciones de Terminales y No Terminales, concretando este análisis en el analizador Léxico siguiente.

### Expresiones Regulares

```
WHITESPACE = [ \t\r\n]+  
ID = [a-zA-Z_][a-zA-Z0-9_]*  
INT = [0-9]+  
DEC = [0-9]+\.+[0-9]+  
STRING = \"([^\\"\\]|\\.)*\"  
HEX = H[0-9A-Fa-f]{6}
```

### Palabras Reservadas

```
"INICIO"           { return symbol(sym.INICIO); }  
"FIN"              { return symbol(sym.FIN); }  
"SI"               { return symbol(sym.SI); }  
"ENTONCES"         { return symbol(sym.ENTONCES); }  
"FINSI"            { return symbol(sym.FINSI); }  
"MIENTRAS"         { return symbol(sym.MIENTRAS); }  
"HACER"             { return symbol(sym.HACER); }  
"FINMIENTRAS"      { return symbol(sym.FINMIENTRAS); }  
"VAR"              { return symbol(sym.VAR); }  
"MOSTRAR"           { return symbol(sym.MOSTRAR); }  
"LEER"              { return symbol(sym.LEER); }
```

### Configuraciones

```
"%"?"DEFAULT"        { return symbol(sym.CONF_DEFAULT); }  
"%"?"COLOR_TEXTO_SI" { return  
symbol(sym.CONF_COLOR_TEXTO_SI); }  
"%"?"COLOR_SI"        { return symbol(sym.CONF_COLOR_SI); }  
"%"?"FIGURA_SI"       { return  
symbol(sym.CONF FIGURA_SI); }  
"%"?"LETRA_SI"        { return symbol(sym.CONF_LETRA_SI); }
```

```

"%"?"LETRA_SIZE_SI"           { return
symbol(sym.CONF_LETRA_SIZE_SI); }
"%"?"COLOR_TEXTO_MIENTRAS"   { return
symbol(sym.CONF_COLOR_TEXTO_MIENTRAS); }
"%"?"COLOR_MIENTRAS"         { return
symbol(sym.CONF_COLOR_MIENTRAS); }
"%"?"FIGURA_MIENTRAS"        { return
symbol(sym.CONF FIGURA_MIENTRAS); }
"%"?"LETRA_MIENTRAS"         { return
symbol(sym.CONF_LETRA_MIENTRAS); }
"%"?"LETRA_SIZE_MIENTRAS"    { return
symbol(sym.CONF_LETRA_SIZE_MIENTRAS); }
"%"?"COLOR_TEXTO_BLOQUE"     { return
symbol(sym.CONF_COLOR_TEXTO_BLOQUE); }
"%"?"COLOR_BLOQUE"           { return
symbol(sym.CONF_COLOR_BLOQUE); }
"%"?"FIGURA_BLOQUE"          { return
symbol(sym.CONF FIGURA_BLOQUE); }
"%"?"LETRA_BLOQUE"            { return
symbol(sym.CONF_LETRA_BLOQUE); }
"%"?"LETRA_SIZE_BLOQUE"       { return
symbol(sym.CONF_LETRA_SIZE_BLOQUE); }

```

## Figuras

```

"ELIPSE"                  { return symbol(sym.FIG_ELIPSE); }
"CIRCULO"                 { return symbol(sym.FIG_CIRCULO); }
"PARALELOGRAMO"           { return symbol(sym.FIG_PARALELOGRAMO);
}
"RECTANGULO"               { return symbol(sym.FIG_RECTANGULO); }
"ROMBO"                    { return symbol(sym.FIG_ROMBO); }
"RECTANGULO_REDONDEADO"   { return
symbol(sym.FIG_RECTANGULO_REDONDEADO); }

```

## Fuente

```

"ARIAL"                     { return symbol(sym.FONT_ARIAL); }
"TIMES_NEW_ROMAN"           { return
symbol(sym.FONT_TIMES_NEW_ROMAN); }
"COMIC_SANS"                 { return symbol(sym.FONT_COMIC_SANS); }
"VERDANA"                   { return symbol(sym.FONT_VERDANA); }

```

## Operadores Relacionales

```
"==" { return symbol(sym.IGUALDAD); }
"!=" { return symbol(sym.DIFERENTE); }
">=" { return symbol(sym.MAYOR_IGUAL); }
"<=" { return symbol(sym.MENOR_IGUAL); }
">" { return symbol(sym.MAYOR); }
"<" { return symbol(sym.MENOR); }
```

## Operadores Lógicos

```
"&&" { return symbol(sym.AND); }
"||" { return symbol(sym.OR); }
"!" { return symbol(sym.NOT); }
```

## Operadores aritméticos

```
"+" { return symbol(sym.MAS); }
"-" { return symbol(sym.MENOS); }
"*" { return symbol(sym.POR); }
"/" { return symbol(sym.DIV); }
```

## Símbolos Especiales

```
"=" { return symbol(sym.ASIG); }
"(" { return symbol(sym.PAR_IQZ); }
")" { return symbol(sym.PAR_DER); }
"," { return symbol(sym.COMA); }
"|" { return symbol(sym.PIPE); }
```

## Análisis De Gramática para Parser

Para el análisis de la gramática se tuvo en cuenta el análisis previo del Lexer, se tomó en cuenta la estructura de la gramática, las producciones que puede tener el lenguaje y algunos errores que esta puede producir concretando este análisis en el analizador Sintáctico siguiente.

### Terminales:

Estos representan las derivaciones que no tienen ninguna otra derivación son estados donde son tokens que ya vienen del análisis Léxico.

```
/*Terminales*/\n\nterminal INICIO, FIN, VAR, SI, ENTONCES, FINSI, MIENTRAS,\nHACER, FINMIENTRAS, MOSTRAR, LEER;\nterminal SEP_SECCIONES;\nterminal MAS, MENOS, POR, DIV, ASIG, PAR_IZQ, PAR_DER;\nterminal IGUALDAD, DIFERENTE, MAYOR, MENOR, MAYOR_IGUAL,\nMENOR_IGUAL;\nterminal AND, OR, NOT;\nterminal PIPE, COMA;\nterminal CONF_DEFAULT, CONF_COLOR_TEXTO_SI, CONF_COLOR_SI,\nCONF FIGURA_SI, CONF_LETRA_SI, CONF_LETRA_SIZE_SI;\nterminal CONF_COLOR_TEXTO_MIENTRAS, CONF_COLOR_MIENTRAS,\nCONF FIGURA_MIENTRAS, CONF_LETRA_MIENTRAS,\nCONF_LETRA_SIZE_MIENTRAS;\nterminal CONF_COLOR_TEXTO_BLOQUE, CONF_COLOR_BLOQUE,\nCONF FIGURA_BLOQUE, CONF_LETRA_BLOQUE, CONF_LETRA_SIZE_BLOQUE;\nterminal FIG_ELIPSE, FIG_CIRCULO, FIG_PARALELOGRAMO,\nFIG_RECTANGULO, FIG_ROMBO, FIG_RECTANGULO_REDONDEADO;\nterminal FONT_ARIAL, FONT_TIMES_NEW_ROMAN, FONT_COMIC_SANS,\nFONT_VERDANA;\nterminal String ID, CADENA, ENTERO, DECIMAL, HEX;
```

### No terminales:

Estos representan estados que generan una lista de derivaciones. Por lo tanto estos permiten la generación de árboles sintácticos para poder analizar la estructura base de aceptación.

```
/* No Terminales */
non terminal programa, seccion_algoritmo, seccion_config;
non terminal lista_instrucciones, instrucion;
non terminal dec_var, asig_var, sent_si, sent_mientras,
sent_mostrar, sent_leer;
non terminal condicion, expresion_relacional;
non terminal expresion;
non terminal lista_configuraciones, instrucion_config;
non terminal figura_val, font_val, color_val;
```

### Precedence:

Indica el valor que tiene la mayor precedencia, es decir que mientras más arriba este declarado (antes de la siguiente instrucción), el parser seguirá el camino que este antes y si en dado caso no cumple aún sigue bajando en su árbol sintáctico o mas conocido como **AST**.

```
/* Precedencia de Operadores */

precedence left OR;
precedence left AND;
precedence left IGUALDAD, DIFERENTE, MAYOR, MENOR,
MAYOR_IGUAL, MENOR_IGUAL;
precedence left MAS, MENOS;
precedence left POR, DIV;
precedence right NOT;
precedence nonassoc PAR_IZQ, PAR_DER;
```

### VISUALIZACION DE LA PRECEDENCIA:

**Nivel 1 (más débil) →**  
**|| (OR lógico) Nivel 2 →**  
**&& (AND lógico) Nivel 3**  
**→ == != > < >= <= (relacionales) Nivel 4**  
**→ + - (suma / resta) Nivel 5**  
**→ \* / (multiplicación / división) Nivel 6 (más fuerte)**  
**→ ! (NOT lógico, unario)**

### VISUALIZACION DEL ARBOL SINTACTICO:

```

start with programa;

programa ::= seccion_algoritmo SEP_SECCIONES seccion_config
           | seccion_algoritmo
           ;

seccion_algoritmo ::= INICIO lista_instrucciones FIN
                     | INICIO FIN
                     ;

lista_instrucciones ::= lista_instrucciones instrucion
                      | instrucion
                      ;

instrucion ::= dec_var
              | asig_var
              | sent_si
              | sent_mientras
              | sent_mostrar
              | sent_leer
              ;

dec_var ::= VAR ID
          | VAR ID ASIG expresion
          ;

asig_var ::= ID ASIG expresion ;

sent_si ::= SI PAR_IZQ condicion PAR_DER ENTONCES
          lista_instrucciones FINSI ;

sent_mientras ::= MIENTRAS PAR_IZQ condicion PAR_DER HACER
                 lista_instrucciones FINMIENTRAS ;

```

```
sent_mostrar ::= MOSTRAR CADENA
                | MOSTRAR ID
                | MOSTRAR expresion
                ;
sent_leer ::= LEER ID ;
expresion ::= expresion MAS expresion
            | expresion MENOS expresion
            | expresion POR expresion
            | expresion DIV expresion
            | PAR_IZQ expresion PAR_DER
            | ENTERO
            | DECIMAL
            | ID
            ;
condicion ::= condicion AND condicion
            | condicion OR condicion
            | NOT condicion
            | expresion_relacional
            | PAR_IZQ condicion PAR_DER
            ;
expresion_relacional ::= expresion IGUALDAD expresion
                        | expresion DIFERENTE expresion
                        | expresion MAYOR expresion
                        | expresion MENOR expresion
                        | expresion MAYOR_IGUAL expresion
                        | expresion MENOR_IGUAL expresion
                        ;
seccion_config ::= lista_configuraciones ;
lista_configuraciones ::= lista_configuraciones
instruccion_config
                    | instruccion_config
                    ;
instruccion_config ::= CONF_DEFAULT ASIG expresion
                    | CONF_DEFAULT ASIG expresion PIPE
expresion
```

```

| CONF_COLOR_TEXTO_SI ASIG color_val
| CONF_COLOR_TEXTO_SI ASIG color_val PIPE
expression
| CONF_COLOR_SI ASIG color_val
| CONF_COLOR_SI ASIG color_val PIPE
expression
| CONF FIGURA_SI ASIG figura_val
| CONF FIGURA_SI ASIG figura_val PIPE
expression
| CONF LETRA_SI ASIG font_val
| CONF LETRA_SI ASIG font_val PIPE
expression
| CONF LETRA_SIZE_SI ASIG expresion
| CONF LETRA_SIZE_SI ASIG expresion PIPE
expression
| CONF COLOR_TEXTO_MIENTRAS ASIG
color_val
| CONF COLOR_TEXTO_MIENTRAS ASIG
color_val PIPE expression
| CONF COLOR_MIENTRAS ASIG color_val
| CONF COLOR_MIENTRAS ASIG color_val PIPE
expression
| CONF FIGURA_MIENTRAS ASIG figura_val
| CONF FIGURA_MIENTRAS ASIG figura_val
PIPE expression
| CONF LETRA_MIENTRAS ASIG font_val
| CONF LETRA_MIENTRAS ASIG font_val PIPE
expression
| CONF LETRA_SIZE_MIENTRAS ASIG expresion
| CONF LETRA_SIZE_MIENTRAS ASIG expresion
PIPE expression
| CONF COLOR_TEXTO_BLOQUE ASIG color_val
| CONF COLOR_TEXTO_BLOQUE ASIG color_val
PIPE expression
| CONF COLOR_BLOQUE ASIG color_val
| CONF COLOR_BLOQUE ASIG color_val PIPE
expression
| CONF FIGURA_BLOQUE ASIG figura_val
| CONF FIGURA_BLOQUE ASIG figura_val PIPE
expression
| CONF LETRA_BLOQUE ASIG font_val
| CONF LETRA_BLOQUE ASIG font_val PIPE
expression
| CONF LETRA_SIZE_BLOQUE ASIG expresion

```

```
| CONF_LETRA_SIZE_BLOQUE ASIG expresion  
PIPE expresion  
;  
  
color_val ::= expresion COMA expresion COMA expresion  
| HEX  
;  
  
figura_val ::= FIG_ELIPSE  
| FIG_CIRCULO  
| FIG_PARALELOGRAMO  
| FIG_RECTANGULO  
| FIG_ROMBO  
| FIG_RECTANGULO_REDONDEADO  
;  
  
font_val ::= FONT_ARIAL  
| FONT_TIMES_NEW_ROMAN  
| FONT_COMIC_SANS  
| FONT_VERDANA  
;
```