**Expresiones Regulares:**

digit → [0-9]

ctei →  digit+

ctef → (ctel\.(ctel)

letter → [a-z A-Z]

id → letter (letter | ctel)

espacios → (tab | blank| newline)

cte.string → ( letter\*| ctef\* | ctel\*| espacios\*)+

char → char

float → float

int → int

var→var

programa→programa

module → module

void → void

return → return

paleta → red | blue | black | green | pink | yellow

; → \;

: → \:

if → if

else → else

main → main

to →to

do →do

relop → > | < | <> | ==

+ → \+

- → \-

\*→ \\*

/→\/

{→\{

}→\}

(→\(

) →\)

“ → \”

write → write

read → read

size →size

color → color

clear → clear

pendown → pendown

penup → penup

arc →arc

circle → circle

point → point

line → line

**Gramatica:**

Programa → programa id ; vars Programa2

Programa2 → Var Programa2 | Programa3

Programa3 → Function Programa3 | main { Estatutos }

Function → Fun | FunVoid | Size | Color | Clear | Pendown | Penup | Arc | Circle | Point | Line

Fun → Tipo module id ( Fun4 ) Fun2 { Fun3 } Return }

Fun2 → Var | ɛ

Fun3 → Estatutos | ɛ

Fun4 → Parametros | ɛ

FunVoid → void module id ( FunVoid4 ) FunVoid2 { FunVoid3 } }

FunVoid2 → Var | ɛ

FunVoid3 → Estatutos | ɛ

FunVoid4 → Parametros | ɛ

Estatutos → Estatutos1 | Estatutos2 | Estatutos3 | Estatutos4 | Estatutos5 | Estatutos6 | Estatutos7

Estatutos1 → Asignacion Estatutos1 | Estatutos | ɛ

Estatutos2 → CallVoid Estatutos2 | Estatutos | ɛ

Estatutos3 → Read Estatutos3 | Estatutos | ɛ

Estatutos4 → Write Estatutos4 | Estatutos | ɛ

Estatutos5 → Decision Estatutos5 | Estatutos | ɛ

Estatutos6 → For Estatutos6 | Estatutos | ɛ

Estatutos7 → While Estatutos7 | Estatutos | ɛ

Asignacion → id = Expresión | id = Function

CallVoid → id ( parameters ) ;

Decision → if ( Expresión ) then { Decision2 ; } Decision1

Decision1 → else { Decision2 ; } | ɛ

Decision2 → Estatutos | ɛ

Read → read ( id Read2 ) ;

Read2 → , id Read2 | ɛ

Write → write ( Write2 ) ;

Write2 → “ cte.string “ Write3 | Expresion Write3

Write3 → , “ cte.string “ Write2 | , Expresion Write2 | ɛ

For → for id = Exp to Exp do { For2 } ; }

For2 → Estatutos | ɛ

While → while ( Expresion ) do { While2 ; }

While2 → Expresion | ɛ

Color → color ( paleta ) ;

Clear → clear ( ) ;

Pendown → pendown ( ) ;

Penup → penup ( ) ;

Arc → arc ( Exp , Exp ) ;

Circle → circle ( Exp ) ;

Point → point ( Exp , Exp ) ;

Size → size ( Exp ) ;

Line → line ( Exp , Exp ) ;

Expresion → Exp relop Exp

Return → return ( exp )

Exp → Termino Exp2

Exp2 → + Termino Exp2 | - Termino Exp 2 | ɛ

Termino → Factor Termino2

Termino2 → \* Factor Termino2 | / Factor Termino2 | ɛ

Parámetros → Tipo id Parametros2

Parametros2 → , id | ɛ

Factor → ( Expresión ) | + VarNum | - VarNum | Varnum

VarNum → id | ctei | ctef

Var→ var Tipo : id ; | var Tipo : id Var2 ;

Var2 → , id Var2 | ɛ

Tipo → int | float | char

Diagramas:













