Proyecto 1

Lenguajes Formales y Autómatas

En la primera fase del proyecto es necesario la lectura de un archivo de texto llamado: GRAMATICA.txt el cual contiene la definición de la gramática.

Dicho archivo está compuesto de las siguientes partes:

1. SETS: Contiene la definición abreviada de un conjunto de símbolos terminales, esta parte puede o no venir dentro del archivo, no es necesario que aparezca, pero si aparece, debe poseer al menos un SET.
    a. Ejemplo

```
SETS
    LETRA   = 'A'..'Z'+'a'..'z'+'_'
    DIGITO  = '0'..'9'
    CHARSET = CHR(32)..CHR(254)
```

    b. Tomar en cuenta las siguientes características:
        i. La palabra SETS debe estar en mayúscula.
        ii. Los sets pueden estar concatenados a través del signo "+", como muestra el set: LETRA.
        iii. Se puede utilizar la función CHR como lo muestra el set: CHARSET.
        iv. Puede haber muchos espacios en blanco entre el identificador, el símbolo "=" y la definición.
        v. Puede haber varios saltos de línea (Enters) entre un SET y otro.
2. TOKENS: Los tokens representan los símbolos terminales y no terminales de la gramática, en esta fase no nos importa si un identificador ha sido declarado o no en los SETS,
    a. Ejemplo

```
TOKENS
    TOKEN 1= DIGITO DIGITO *
    TOKEN  51  = ':'
    TOKEN 3= LETRA ( LETRA | DIGITO )*   {
RESERVADAS() }
```

    b. LA PALABRA TOKENS debe existir y estar en mayúscula
    c. Esta sección debe existir
    d. Cada token debe poseer la palabra: TOKEN y un número, seguido del signo igual "=".
    e. Después del signo igual debe venir una expresión regular, que puede ser uno o varios caracteres (Encerrados en apóstrofes).
    f. Los signos utilizados para las operaciones de las expresiones regulares son los únicos que no necesitan estar entre comillas, a menos que se quiera denotar su uso como signo terminal.
        i. Los signos de operaciones para las expresiones regulares son: + * ? ( ) |
3. ACTIONS: La palabra ACTIONS contiene definición de funciones, en este caso específico las palabras reservadas del lenguaje, es importante que la función: Reservadas() siempre debe existir y puede haber otras funciones.

a. Ejemplo

```
ACTIONS
RESERVADAS()
{
        18 = 'PROGRAM'
        19 = 'INCLUDE'
        20 = 'CONST'
        39 = 'DOWNTO'
}
```

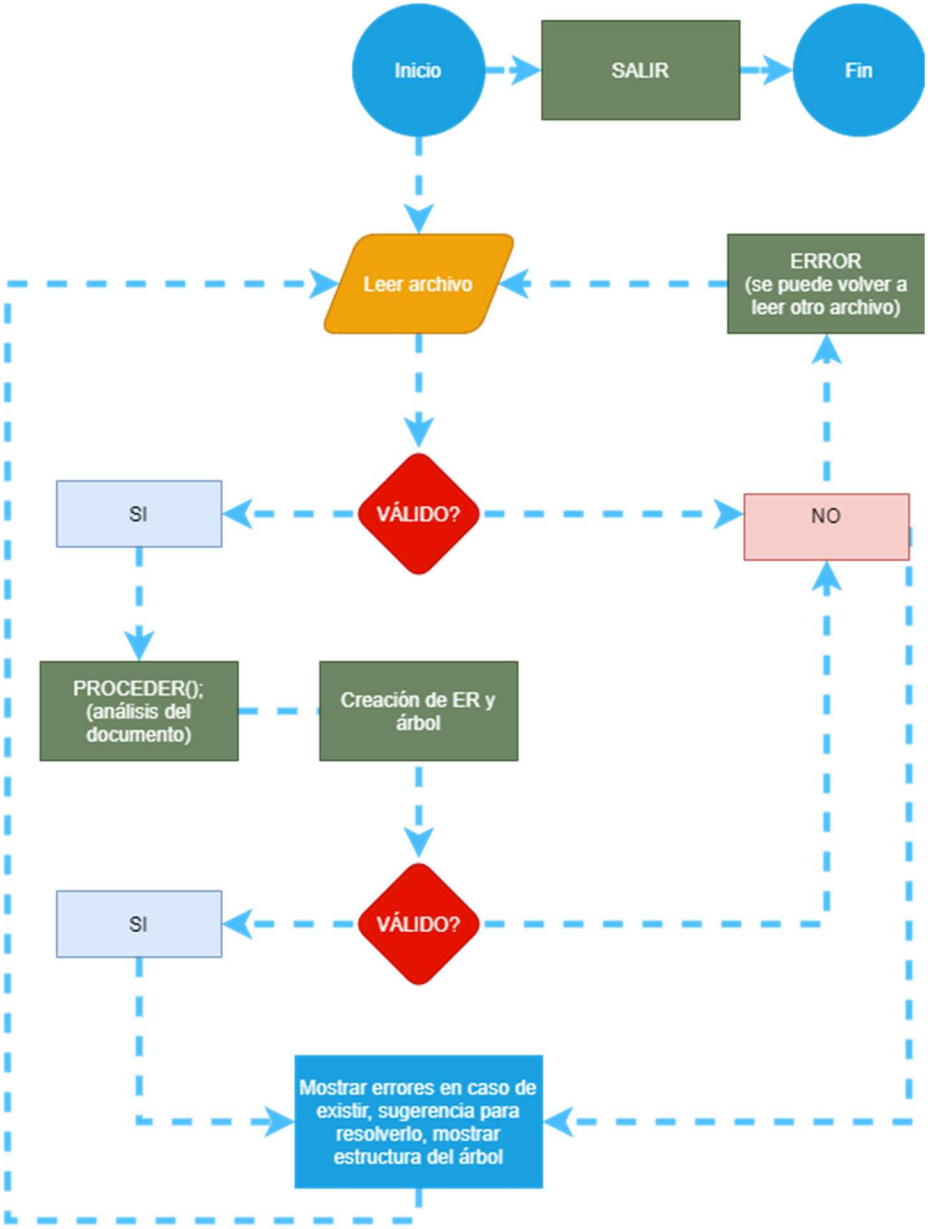
b. La palabra ACTIONS siempre debe venir acompañada de la función RESERVADAS ().
c. Todas las funciones deben tener un identificador y unos paréntesis abierto y cerrado.
d. Las funciones descritas en ACTIONS deben iniciar y finalizar con llaves {}.
e. Los tokens dentro están conformados por: número, signo igual y luego el identificador entre apóstrofes

4. ERROR: La definición de errores debe venir al menos uno, el ERROR debe tener asignado un número, y el identificador debe tener como sufjio la palabra ERROR en mayúscula:
    a. Ejemplo:

```
ERROR = 54
```

    b. Los identificadores solo deben poseer letras, y en la parte derecha del símbolo igual, solamente puede haber números.

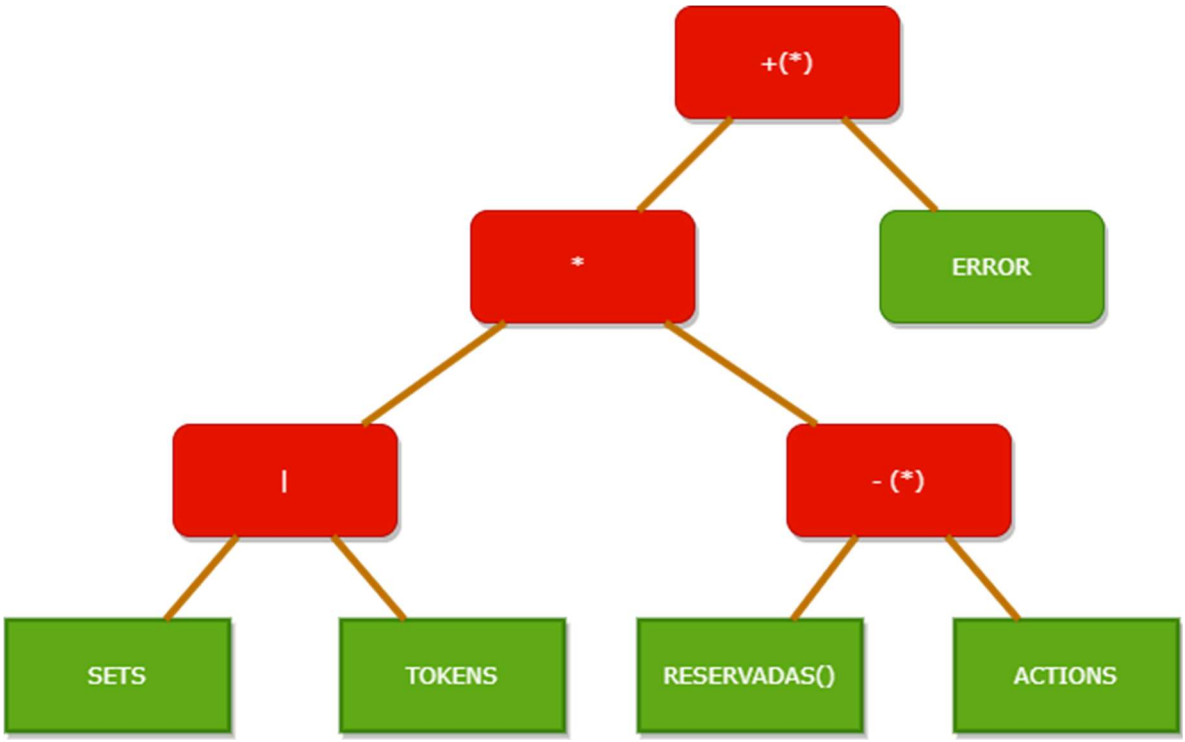
Diagrama de flujo



Expresiones regulares:

```
Terminales =
                    "((?i)SETS|TOKENS|ACTIONS(?-i))([\W \W]+|)$")
Variables =
      "(([A-Za-z])+( *)=(( |)*(( *|(\+))(')(.)(')(..)(')(.)(')) )*)*|(( *|\+?
         *)(')(.)(')(..)(')(.)(')( )*(\+)( )*(')(.)(')(..)(')(.)(')( )*)*|((
  *|(\+))(')(.)(')( *))*|( |)*(CHR)(\()[0-9]+(\))..(CHR)(\()[0-9]+(\))( |)*)*(|)$)")
Tokens =
"(TOKEN)( |   )*(\d)+( |    )*=( | )*((\(+)( |  )*[A-Z]+( |   )*[A-Z]+( |   )*(\)+)( |
              )*)+|(( |    |)*(')+(.)+(')+)+( |      |)*)+)|(([A-Z]*)+( |
             )*((\*|\||\(|\)|\{|\}))+)( |      ))*|([A-Z]+( |      )*[A-Z]+( |
                           )*((\*|\||\(|\)|\{|\}))|\})+")
Funciones =
                   @"(([A-Z])+(\(\))( )*)|( )*((\{)|(\}))( )*");
Errores =
                         @"(ERROR)( )*=( )*([0-9]+)( )*");
Tokens Funciones
               @"(([0-9])+( )*=( )*(')([A-Z]+)(')( *|))|( )*((\{)|(\}))( )*");
```

Árbol (GRAMATICA.EXE)




```
"ERROR","+","ACTIONS","-","RESERVADAS()","*","TOKENS","|","SETS"
```