Desarrollo de analizador léxico

Jaime Sáez de Buruaga Brouns

1. Clases del lenguaje

- Programa: esta clase define el lenguaje: genera todas las palabras posibles pertenecientes al lenguaje.
- Declaraciones: define todas las posibles declaraciones que pueden darse en el lenguaje.
- Instrucciones: define todas las posibles instrucciones posibles que pertenecen al lenguaje.
- Tipo: esta clase representa todos los tipos posibles de las variables definidas en el lenguaje (int, real o bool).
- Variable: esta clase genera todos los posibles nombres que puede tener una variable en nuestro lenguaje.
- Letra: esta clase representa todas las letras posibles que aparecen en el lenguaje (alfabeto).
- Carácter: esta clase representa todos los caracteres posibles que pueden aparecer en nuestro lenguaje (a-z, 0-9, _).
- Dígito: esta clase genera todos los dígitos posibles que pueden aparecer en el lenguaje (0-9).
- Asignación: esta clase genera todas las posibles asignaciones que pueden existir, en nuestro caso solo asignaciones variable = expresión.
- Expresión: genera todos los posibles valores que pueden ser asignados a una expresión (un número entero o real, una variable o un booleano).
- Entero: genera todo el conjunto posible de números enteros.
- DígitoPositivo: genera un dígito positivo (1-9).
- Real: genera todo el conjunto posible de números reales.
- Decimal: genera todo el conjunto posible de parte decimal (0.0000..0.99999).
- Exponencial: genera todo el conjunto posible de parte exponencial (e^X).
- ExpCompleja: genera todo el conjunto posible de expresiones del lenguaje.
- Operador: genera todos los operadores posibles pertenecientes al lenguaje (aritméticos binarios, lógicos y relacionales).

2. Especificación formal del léxico mediante definiciones regulares

DUDA: En los reales dice “no se permite la aparición de 0s no significativos a la derecha”, pero en el ejemplo sale -002.00, que además también rompe con “en los enteros no se permite la aparición e 0s no significativos a la izquierda”.

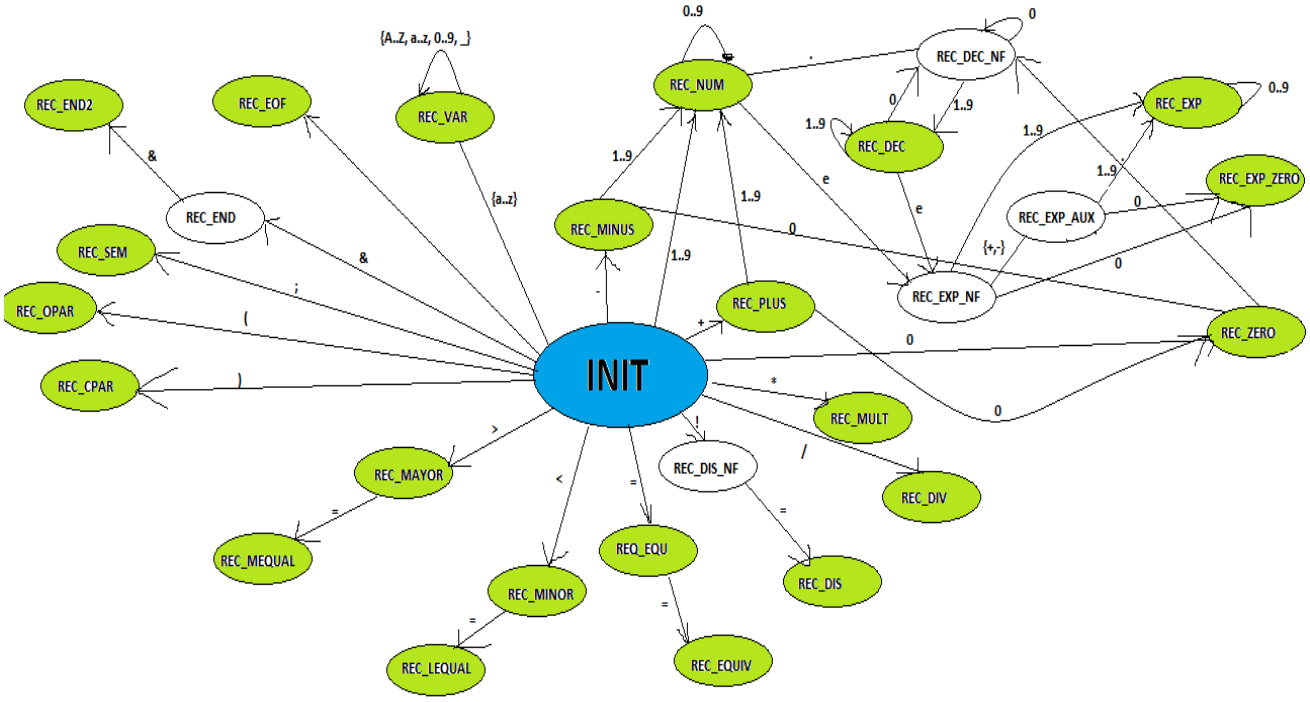
- **Definiciones auxiliares**

Dígito -> 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
Letra -> a | b | c | d | e | f | g | h | i | j | k | l | m | n | ñ | o | p | q | r | s | t | u | v | w | x | y | z
DígitoPositivo -> 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
Carácter -> Letra | Dígito | _
Tipo -> int | real | bool
Expresión -> Entero | Real | Variable | true | false | ExpCompleja

- **Definiciones léxicas**

Programa ->Sec_Declaraciones && Sec_Instrucciones
Sec_Declaraciones -> Declaración (;Declaración)*
Sec_Instrucciones -> Asignación(;Asignación)*
Asignación -> Variable = ExpCompleja
ExpCompleja -> ExpComplejaOperadorExpCompleja | (ExpCompleja) | Expresión
Declaración -> Tipo Variable
Variable -> Letra(Carácter*)
Operador -> + | - | * | / | (-) | and | or | not | < | > | <= | >= | == | !=
Entero -> (+ | -)(DígitoPositivo)Dígito*
Real -> (+ | -)Entero(Decimal) | (+ | -)Entero(Exponencial) | Decimal(Exponencial)
Decimal -> .(Dígito*)DígitoPositivo
Exponencial -> e(Entero)

3. Diseño de analizador léxico mediante diagrama de transiciones.



NOTA: Los estados denotados con NF (blancos) son de no aceptación, mientras que los verdes son estados de aceptación.
NOTA2: Para probar correctamente las implementaciones es necesario introducir los ficheros “pruebas/input{1,2,3}.txt” en el directorio en el que se invoca la ejecución, que debe ser el mismo de los archivos fuente *.java.