Manual Tecnico

Programa de Analisis Lexico y Sintactico

Fernando José Rodríguez Ramírez

Noviembre, 2021

Introducción

El objetivo del presente manual es que el usuario o programador tenga un mejor entendimiento del funcionamiento interno del programa y sus componentes, para posterior mantenimiento o ampliación.

Analizador Léxico

Concepto: Para realizar el análisis léxico del texto, el programa utiliza una Máquina de Mealy, siendo este un Autómata Finito Determinista que es capaz de realizar acciones en sus transiciones, esto con el objetivo de realizar la tokenización, junto con un buffer para almacenar caracteres en su recorrido para convertirlos en los lexemas de los tokens.

Uso: Para la creación de la máquina de Mealy se debe de definir un autómata finito determinista, para esto se declara la función transición del autómata. Cada transición es un objeto que almacena el estado de donde se parte, el estado a dónde se dirige, los caracteres que hacen a la transición, y las acciones que se llevarán a cabo. Estas transiciones están organizadas en una matriz de transiciones.

La máquina de Mealy del programa cuenta con 3 acciones básicas para realizar entre transiciones respecto al buffer.

- Guardar: Que obtiene los caracteres almacenados en el buffer y los guarda en un token.
- Agregar: Agrega el carácter actual al buffer.
- Vaciar: Vacía el buffer

Estas acciones se pueden combinar ya que las transiciones cuentan con un arreglo de acciones a realizar.

```
new Transicion('A','E',Alfabeto.SIMBOLOS_UNITARIOS,new AccionesEnum[]{AccionesEnum.AGREGAR,AccionesEnum.GUARDAR}),
new Transicion('A','X',Alfabeto.SIMBOLOS_LITERALES,AccionesEnum.AGREGAR),
new Transicion('A','A',Alfabeto.ESPECIAL,new AccionesEnum[]{})},
```

Consejo: Las acciones en el arreglo deben de estar en el orden en el que se quiere que sean ejecutadas.

El autómata se guiará por las transiciones creadas para tokenizar el texto ingresado.

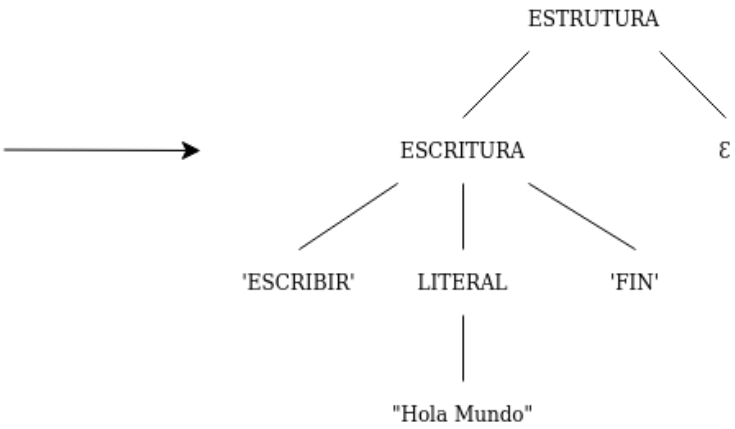
```
static Transicion[][] transiciones = {
  {new Transicion('A','B',Alfabeto.DIGITOS,AccionesEnum.AGREGAR),
  new Transicion('A','D',Alfabeto.LETRAS,AccionesEnum.AGREGAR),
  new Transicion('A','H','*',AccionesEnum.AGREGAR),
  new Transicion('A','F','/',AccionesEnum.AGREGAR),
  new Transicion('A','D','_',AccionesEnum.AGREGAR),
  new Transicion('A','C','-',AccionesEnum.AGREGAR),
  new Transicion('A','E',Alfabeto.SIMBOLOS_UNITARIOS,new AccionesEnum[]{AccionesEnum.AGREGAR,AccionesEnum.GUARDAR}),
  new Transicion('A','X',Alfabeto.SIMBOLOS_LITERALES,AccionesEnum.AGREGAR),
  new Transicion('A','A',Alfabeto.ESPECIAL,new AccionesEnum[]{}),

  {new Transicion('B','B',Alfabeto.DIGITOS,AccionesEnum.AGREGAR),
  new Transicion('B','X',Alfabeto.LETRAS,AccionesEnum.AGREGAR),
  new Transicion('B','H','*',new AccionesEnum[]{AccionesEnum.GUARDAR,AccionesEnum.AGREGAR}),
  new Transicion('B','F','/',new AccionesEnum[]{AccionesEnum.GUARDAR,AccionesEnum.AGREGAR}),
  new Transicion('B','X','_',AccionesEnum.AGREGAR),
  new Transicion('B','X','-',AccionesEnum.AGREGAR),
  new Transicion('B','E',Alfabeto.SIMBOLOS_UNITARIOS,new AccionesEnum[]{AccionesEnum.GUARDAR,AccionesEnum.AGREGAR,AccionesEnum.GUARDAR}),
  new Transicion('B','X',Alfabeto.SIMBOLOS_LITERALES,AccionesEnum.AGREGAR),
  new Transicion('B','A',Alfabeto.ESPECIAL,AccionesEnum.GUARDAR)},
};
```

Forma de Declarar las Transiciones

Analisis Sintactico

Concepto: Para realizar el análisis sintáctico el programa utiliza un autómata de pila que estructura los tokens obtenidos del análisis sintáctico en un árbol de parseo.

Tokens	
IDENTIFICADOR	ESCRIBIR
LITERAL	"Hola Mundo"
IDENTIFICADOR	FIN



Para realizar el parseo el analizador solo requiere de la gramática, esta se traduce en código Java por medio de reglas de producción, siendo estas objetos que contienen el nombre de la producción inicial y un arreglo con las diferentes producciones a las que lleva.

```
new ReglaProduccion(
    "ESTRUCTURA",
    new String[][]{
        {"ESCRITURA","ESTRUCTURA"},
        {"E_REPETIR", "ESTRUCTURA"},
        {"CONDICIONAL", "ESTRUCTURA"},
        {"E_MATEMATICA", "ESTRUCTURA"},
        {""}
    }
),
```

El autómata de pila está diseñado para no necesitar más que la gramática para su funcionamiento.

Consejo: Ninguna regla de producción puede llamarse igual que un no terminal.