Universidad Simon Bolivar
Departamento de Computacion y Tecnologia de la Informacion
CI3725 - Traductores e Interpretadores
Octubre – Enero 2014

Lexer – Analizador Lexicografico
Brainiac

Integrante:
David Lilue 09-10444

Implementacion

Antes que nada el lenguaje que se decidio utilizar es “Ruby” a pesar de no poseer una herramiente que analice la parte lexicografica. Se tuvo que expresar todas las expresiones regulares necesarias para poder derivar todos lo tokens que representan cada uno de las frases pertenecientes al lengueaje. Ademas de crear las clases fundamentales diferenciando los tokens de los errores lexicos, tambien crear subclases para cada una de las expresiones regulares cada una con el nombre Tk<nombre> a patir de la clase general “PhraseS” que posee la linea, columna y texto de algo; de esta se deriva tambien “LexicographError” que es para los errores.
La clase mas importante es “Lexer” que es donde se realiza todo el recorrido del archivo de entrada ademas de la distincion de los tokens y errores. Guardandolos en arreglos separados para luego ser impreso tal como se especifico en el enunciado.

Revision Teorico-Practica

1. E1 = (else) , E2 =(end) , E3 = (a,b,..,z,A,B,..,Z,_)(a,b,..,z,A,B,..,Z,0,1,..,9,_)*
2.
    1. else

	e	l	s
q1	q2	{}	{}
q2	{}	q3	{}
q3	{}	{}	q4
q4	qf	{}	{}
qf	{}	{}	{}

    2. End

	e	n	d
q1	q2	{}	{}
q2	{}	q3	{}
q3	{}	{}	qf
qf	{}	{}	{}

    3. Ident
    a = (a,b,..,z,A,B,..,Z,_)
    b = (a,b,..,z,A,B,..,Z,0,1,..,9,_)

	a	b
q1	qf	{}
qf	{}	qf

3.

	e	l	s	n	d	a	b	landa
q0	{}	{}	{}	{}	{}	{}	{}	q1,q2,q3
q1	q2	{}	{}	{}	{}	{}	{}	{}
q2	{}	q3	{}	{}	{}	{}	{}	{}
q3	{}	{}	q4	{}	{}	{}	{}	{}
q4	qf1	{}	{}	{}	{}	{}	{}	{}
q5	q6	{}	{}	{}	{}	{}	{}	{}
q6	{}	{}	{}	q7	{}	{}	{}	{}
q7	{}	{}	{}	{}	qf2	{}	{}	{}
q8	{}	{}	{}	{}	{}	qf3	{}	{}
qf1	{}	{}	{}	{}	{}	{}	{}	{}
qf2	{}	{}	{}	{}	{}	{}	{}	{}
qf3	{}	{}	{}	{}	{}	{}	qf3	{}

4. qf1 == E1
   qf2 == E2
   qf3 == E3

5. Existe un conflicto con las palabras end y else, dado que el lenguaje de E3 puede reconocerlo. Por lo tanto:

   else pertenece a L1 y L3
   end pertenece a L2 y L3

6. x = (a,b,..,z,A,B,..,Z,_)
   y = (a,b,..,z,A,B,..,Z,0,1,..,9,_)

	x	y	e	l	s	n	d
q0	qf3	qf3	{q2,q6},qf3	qf3	qf3	qf3	qf3
{q2,q6}	{}	{}	{}	q3	{}	q7	{}
q3	{}	{}	{}	{}	q4	{}	{}
q4	{}	{}	qf1	{}	{}	{}	{}
q7	{}	{}	{}	{}	{}	{}	qf2
qf1	{}	{}	{}	{}	{}	{}	{}
qf2	{}	{}	{}	{}	{}	{}	{}
qf3	{}	qf3	{}	{}	{}	{}	{}

7. Los conflictos se pueden notar, dado que no se puede conseguir un automata deterministico, exite un nodo que con una misma letra puede ir a dos nodos diferentes.

8. <L(E1), L(E2), L(E3)>

Si el estado final es qf1 -> L(E1)
Si el estado final es qf2 -> L(E2)
Si el estado final es qf3 -> L(E3)

9. ....

10. Un caso similar sucedio en el transcurso del desarrollo del lexer, por lo que se tuvo que priorizar el reconocimiento de las palabras, parecido a la pregunta 8. Asi como en otras situaciones parecidas, se tuvo que lidiar con ellas para que se diferenciaran las distintas frases sin confundirse de tokens.