Práctica 1 Analizador Léxico Lenguaje TL

Grupos: Máximo 2 personas por grupo, ambas personas deberán estar presentes el día de la entrega y estar en capacidad de alterar el programa dado que surjan nuevos requerimientos durante la sustentación).

El objetivo para esta práctica es tomar un código fuente escrito en el lenguaje TL y realizar un análisis léxico sobre dicho código.

Fecha de entrega: 12/03/2018

Entrada - Salida

La entrada para juzgar el analizador léxico está dada por un archivo que contiene código fuente escrito en TL, que su programa debe analizar a nivel léxico para entregar la salida adecuada.

La salida consiste en una lista de tokens separados por saltos de línea, los cuales deben seguir el siguiente formato **<tipo_de_token,lexema,fila,columna>** . En el caso de las palabras reservadas, el token y el lexema son iguales. Entonces se debe imprimir el siguiente formato **<tipo_de_token,fila,columna>**. Queda como parte del ejercicio encontrar el conjunto de palabras reservadas para TL, según la especificación dada en el manual de referencia.

Por ejemplo:

Entrada (in01.txt)	Salida (out01.txt)
while for if log funcion	<while,1,1> <for,3,14> <if,4,6> <log,5,8> <funcion,6,1>

NOTAS:

- **log** es una palabra reservada que sirven para escribir en consola. Por su parte, **funcion** es una palabra reservada que sirve para declarar una función.
- Tenga en cuenta que TL es case sensitive, por lo que una cadena como WhiLe no será aceptada como palabra reservada.

Operadores Especiales

Los operadores y símbolos especiales deben imprimirse en el siguiente formato:

<nombre_token,fila,columna>

Donde nombre_token será determinado por la siguiente tabla:

Token	Nombre
{	token_llave_izq
}	token_llave_der
#	token_com
[	token_cor_izq
]	token_cor_der
(	token_par_izq
)	token_par_der

>	token_mayor
<	token_menor
>=	token_mayor_igual
<=	token_menor_igual
in	token_in
==	token_igual_num
.	token_point
!=	token_diff_num
&&	token_and
||	token_or
!	token_not
+	token_mas
-	token_menos
*	token_mul
/	token_div
%	token_mod
^	token_pot
=	token_assign

Se deben especificar los lexemas de la siguiente manera, dependiendo de su token (recordar que los únicos valores que pueden ir fuera de comillas son los numéricos).

<token_string,lexema,fila,columna>
<token_integer,lexema,fila,columna>
<token_float,lexema,fila,columna>

Los identificadores encontrados seguirán el siguiente formato:

<id,lexema,fila,columna>,

Los comentarios deberán ser ignorados.

Errores

Si durante el proceso se detecta un error léxico se deberá abortar el análisis y se debe reportar el error de la siguiente manera:

>>> Error lexico(linea:X,posicion:Y)

Donde X y Y son números enteros que representan el número de línea y la posición donde fue detectado el inicio del error léxico.

Ejemplos

Entrada	**Salida**

log("prueba del if")
a= false
b = true
if a && b {
  log ("El resultado es verdadero")
}
# ignorar comentario 	
log("El valor es falso")
	<log,1,1>
	<token_par_izq,1,4>
	<token_string,prueba del if,1,5>
	<token_par_der,1,20>
	<id,a,2,1>
	<token_assign,2,2>
	<false,2,4>
	<id,b,3,1>
	<token_assign,3,3>
	<true,3,5>
	<if,4,1>
	<id,a,4,4>
	<token_and,4,6>
	<id,b,4,9>
	<token_llave_izq,4,11>
	<log,5,3>
	<token_par_izq,5,7>
	<token_string, El resultado es verdadero,5,8>
	<token_par_der,5,35>
	<token_llave_der,6,1>
	<log,8,1>
	<token_par_izq,8,4>
	<token_string,El valor es falso,8,5>
	<token_par_der,8,24>

- Si en la entrada de texto se detecta una cadena como esta 90.00.50, el analizador deberá detectar un token_float 90.00, luego detectar el token_point, y por último el token_integer 50. Esto ocurre porque el "." podría aparecer como un token único en este lenguaje; de lo contrario, sería un error léxico.
- Si se detecta por ejemplo la cadena 1&23948998, el analizador deberá detectar el entero 1 y luego marcar un error en el &.
- Recuerde tener en cuenta casos extremos como por ejemplo: 2.5598055NOT3!88& , en este caso se detecta la siguiente secuencia:

  <token_float,2.5598055,1,1>
  <id,NOT3,1,10>
  <token_not,1,14>
  <token_integer,88,1,15>
  >>> Error lexico(linea:1,posicion:17)

- A la hora de imprimir una palabra reservada como "log", debe imprimir la misma palabra reservada en el espacio de token <log,fila,columna>.
- A la hora de realizar el análisis léxico se deberá seguir el principio de subcadena más larga, a continuación un ejemplo:

!= **$**	<token_diff_num,1,1> >>> Error lexico(linea:1,posicion:3)

Como se puede ver el análisis toma el token '!=' pues es el token más largo que puede tomar, y con el substring restante '$' informa un error léxico pues este carácter no está definido dentro de los tokens válidos. para el lenguaje.