Implementación de analizadores léxicos

Descripción de la aplicación

Un método para construir un analizador léxico consiste en usar un AFD para describir la conducta de este. Se parte de una tabla descriptiva de los componentes léxicos que va a reconocer el analizador, donde se clasifican los diferentes tipos de componentes.

Posteriormente se construye un AFD para reconocer cada una de las diferentes clases de componentes y se integran en un solo diagrama. Finalmente se determina la matriz de transición de estados.

Una vez definida la matriz de transición de estados la construcción del analizador consiste en aplicar el algoritmo de reconocimiento de cadenas descrito en la sección anterior y añadir un switch/case para tomar acción de acuerdo al tipo de cadena identificada.

Ejemplo:

Construir un analizador léxico que acepte palabras reservadas, identificadores, enteros positivos y operadores aritméticos y relacionales. Además debe eliminar blancos y comentarios y almacenar en una tabla de símbolos los identificadores y las constantes.

Solución:

Palabras reservadas.

El analizador reconocerá las siguientes 20 palabras reservadas:

PROGRAM	END	STEP	TO
ARRAY	INTEGER	CASE	DO
VAR	IF	ELSE	WHILE
CONST	THEN	CHAR	REPEAT
BEGIN	REAL	FOR	UNTIL

y se considera que están almacenadas en las primeras 20 posiciones de la tabla de símbolos. Observe que bajo este supuesto no es relevante de que palabras se trate, ya que la única información que sé es el número total de palabras reservadas.

Tabla de componentes léxicos

La tabla de componentes léxicos incluye tres columnas: componente, clase y tipo.

El tipo de componentes se usa para diferenciar componentes de una misma clase, como en el caso de los operadores aritméticos. Para las palabras reservadas, identificadores y constantes, el tipo especifica la dirección donde se encuentra almacenado el componente en la tabla de símbolos (Dir TS).

La tabla de componentes léxicos

Componente léxico	Clase	Tipo
Palabra reservada	1	Dir TS
Identificadores	2	Dir TS
Constante entera	3	Dir TS
+	4	1
-	4	2
*	4	3
/	4	4
<	5	1
<=	5	2
<>	5	3
>	5	4
>=	5	5
=	5	6
:=	6	0
;	7	0

Descripción de estados de reconocimiento

En la siguiente tabla se indica cual es la función de cada estado de reconocimiento y si en el proceso se esta dejando o no un carácter pendiente de procesar.

Esto ultimo se debe a que hay algunas situaciones en que el reconocimiento del componente se realiza con el ultimo carácter analizado, como es el caso de los componentes de + y <=. Sin embargo en el caso de los componentes de < ó identificador, se requiere leer un carácter adicional para reconocer el componente. En el diagrama del autómata esta situación ocurre en todos los estados de reconocimiento marcado con un asterisco. La información de si ha quedado un carácter pendiente de analizar, se debe pasar al analizador sintáctico para que sea considerada al llamar de nueva cuenta al analizador léxico.

Otros aspectos importantes es la numeración de los estados. Se usan las convenciones siguientes: estados que no son de reconocimiento se numeran del 0 al 99, de reconocimiento de componente se numeran del 100 al 199 y de reconocimiento de error del 200 al 299.

Con este sistema de numeración de estados, se puede determinar que se ha llegado a un estado de reconocimiento cuando este sea mayor o igual a 100. Para luego a través de un switch/case identificar el tipo de componente de que se trata.

Estado de reconocimiento	Componente	Pendiente
100	/	Si
101	+	No
102	-	No
103	*	No
104	<=	No
105	<>	No
106	<	Si
107	>=	No
108	>	Si
109	=	No
110	:=	No
111	identificador	Si
112	Constante entera	Si
113	;	No
200	Se esperaba un =	No
201	Carácter invalido	No

Observe que no incluye un estado de reconocimiento para el caso de palabras reservadas. Esto se debe a que en el caso de los identificadores, antes de reportarlos como de esta clase, se busca en la tabla de símbolo y si se les encuentra en alguno de los primeros 20 lugares se reportan como una palabra reservada.

Matriz de transición de estados

Q= {0,1,2, …, 8, 100,101,…,113,200,201}

∑ = {L, D, +, -, *, /, <,>, =, :, b, ;, otro}

q0 = 0;

F = {100, 101, 102, …, 113, 200,201}

∫	L	D	+	-	*	/	<	>	=	:	b,/n,/t	;	Otro
0	7	8	101	102	103	1	4	5	109	6	0	113	201
1	100	100	100	100	2	100	100	100	100	100	100	100	100
2	2	2	2	2	3	2	2	2	2	2	2	2	2
3	2	2	2	2	2	0	2	2	2	2	2	2	2
4	106	106	106	106	106	106	106	105	104	106	106	106	106
5	108	108	108	108	108	108	108	108	107	108	108	108	108
6	200	200	200	200	200	200	200	200	110	200	200	200	200
7	7	7	111	111	111	111	111	111	111	111	111	111	111
8	112	8	112	112	112	112	112	112	112	112	112	112	112

Analizador léxico

```
main()     /* Controla el ciclo de llamadas al analizador léxico
{
fp=Abrir_archivo("CODIGO:TXT");
c=getchar(fp);
pendiente=1;
while (!EOF)
{

scan()
/* En este punto se tiene la información del símbolo identificado en
componente, clase y tipo. Además, la tabla de símbolo ya se actualizó/

}
}

scan()
{
if(not pendiente) /* si no hay un carácter pendiente de procesar */
     c=getchar(fp) /* extrae el siguiente carácter del archivo */


pendiente=0; componente = " "; q = 0;







while (m[q][c] < 100)
{
componente = componente + c;
c=getchar(fp);
}

switch(q)

{
100: clase = 4; tipo = 4; pendiente = 1;
101: clase = 4; tipo = 1;
102: clase = 4; tipo = 2;
103: clase = 4; tipo = 3;
104: clase = 5; tipo = 2
105: clase = 5; tipo = 3;
```

```
106: clase = 5; tipo = 1; pendiente = 1;
107: clase = 5; tipo = 5;
108: clase = 5; tipo = 4; pendiente = 1;
109: clase = 5; tipo = 6;
110: clase = 6; tipo = 0;
113: clase = 7; tipo = 0;

111:
{
    tipo = guarda(componente);
if(tipo <20)
clase = 1;
else
clase = 2;
pendiente = 1;
}
112: clase = 3; tipo=guarda(componente); pendiente = 1;

200: Error("Se esperaba un =");

201: Error("Carácter invalido"):
}
}
```

Estructura de datos utilizados	
Código fuente	Se introduce en un archivo de texto, desde donde se analiza carácter por carácter

Función de transición de estados	Se mantiene en una matriz m [q] [c].
Tabla de símbolos	En esta tabla se guardan los componentes de tipo palabra reservada, identificador y constante. Todas las palabras reservadas se deben cargar inicialmente en la tabla. Cada componente identificado se debe darse de alta en la tabla y no se debe repetir el registro: Componente, clase y tipo.

Variables Utilizadas	
fp	Apunta al archivo con el código fuente
c	Apunta al carácter que se procesa actualmente
q	Contiene el estado actual
Componente	Cadena en la que se va formando el componente conforme transcurre el análisis.
Pendiente	Indica si no se proceso el ultimo carácter en la llamada anterior
Clase	Contiene la clase del componente identificado
Tipo	Contiene el tipo del componente identificado o su dirección en la tabla de símbolos para las palabras reservadas, identificadores y constantes.

Funciones Utilizadas	
Guarda(componente)	Esta función registra el componente en la tabla de símbolos y regresa la dirección donde lo guardo o donde lo encontró.
Error(mensaje)	Función para generar los mensajes de error.

Ejercicio

1. Construya el analizador léxico descrito en esta sección