Análisis Léxico

Cázares Rodríguez Víctor Manuel
Limones Moscoso Ulises
Ojeda Ávila Diego Antonio
Rivera Arellanes Josué David

Grupo 2
Compiladores
Semestre 2020-2

Análisis del problema

El analizador léxico debe ser capaz de devolver los tokens que encuentre en el archivo de entrada, los cuales serán empleados a su vez en el analizador sintáctico. Aquí tendremos las reglas que definen si se están ingresando los tokens válidos, por lo que debemos especificar su construcción. Si tenemos identificadores debemos definir cómo deben estar construidos, ya sea si deben iniciar por minúsculas o con mayúsculas, si hay números debemos definir también si nuestro lenguaje aceptará solamente números enteros o se podrán usar números de otro tipo. Todas estas reglas se deben especificar en el analizador léxico.

Diseño de la solución
i. Separar los terminales de los no terminales

Terminales
estructura
inicio
fin
ent (entero)
real
dreal
car (caracter)
sin (sin tipo)
[num]
id
def
[
]
si
entonces
sino
mientras
hacer
segun
escribir
leer
devolver
terminar
caso
num
pred
o (or)
y (and)
no
(
)
verdadero

falso
>
<
<=
>=
<>
=
%
cadena
+
-
*
/
. (punto)
;
,
:=

No terminales
programa
declaraciones
tipo_registro
tipo
base
tipo_arreglo
lista_var
funciones
argumentos
lista_arg
arg
tipo_arg
param_arr
sentencias
sentencia
casos
predeterminado
e_bool
relacional
oprel
expresion
oparit
variable
dato_est_sim
arreglo
parámetros
lista_param

ii. Las expresiones regulares para los terminales

estructura→	estructura
inicio→	inicio
fin→	fin
ent →	ent
real→	real
dreal→	dreal
car →	car
sin →	sin
[num]→	\[([0-9]+|[0-9]+.[0-9]+) \]
id→	[a-zA-Z_][a-zA-Z_0-9]*
def→	def
[→	\[
]→	\]
si→	si
entonces→	entonces
sino→	sino
mientras→	mientras
hacer→	hacer
según→	segun
escribir→	escribir
leer→	leer
devolver→	devolver
terminar→	terminar
caso→	caso
num→	([0-9]+|[0-9]+.[0-9]+)
pred→	pred
o →	o
y →	y
no→	no
(→	\(
)→	\)
verdadero→	verdadero
falso→	falso
>→	>
<→	<
<=→	<=
>=→	>=
<>→	<>
=→	=

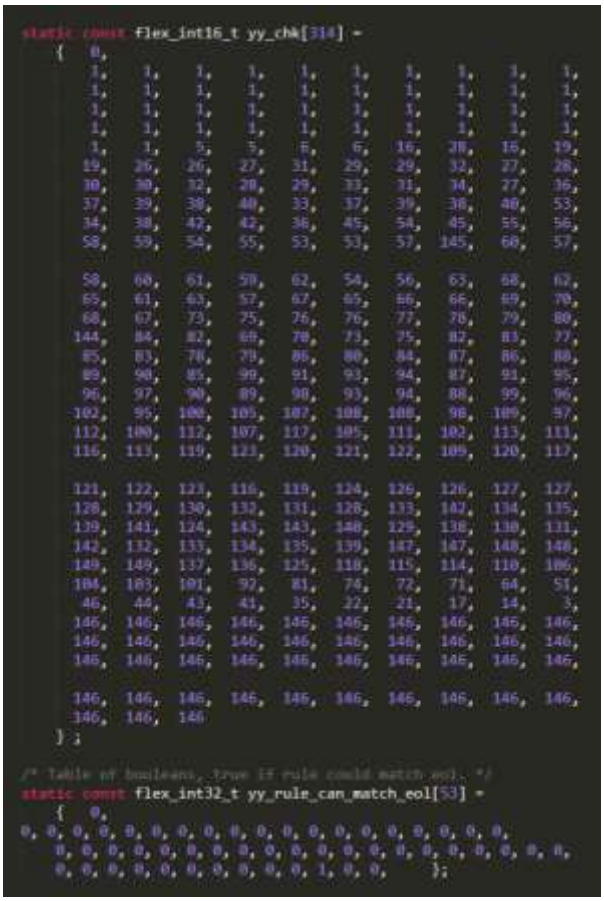
% →	%
Cadena→	[a-zA-Z0-9]*
+→	+
-→	-
*→	*
/→	/
. →	.
;→	;
,→	,
:→	:
:=→	:=

iii. El AFD resultante (Imagen)

```
static const flex_int16_t yy_accept[147] =
    {   0,
       49,   49,   53,   51,   50,   50,   37,   27,   28,   40,
       38,   45,   39,   42,   41,   48,   43,   44,   32,   36,
       31,   47,   10,   11,   47,   47,   47,   47,   47,   47,
       47,   47,   47,   47,   24,   47,   47,   47,   47,   47,
       25,   50,   48,   48,   48,   49,   46,   33,   35,   34,
       47,   47,   47,   47,   47,   47,   47,   47,   47,   47,
       47,   47,   47,   26,   47,   47,   47,   12,   47,   47,
       48,    7,   47,    9,   47,   47,    4,   47,   47,   47,
        3,   47,   47,   47,   47,   47,   47,   47,    8,   47,
       47,   23,   47,   47,   47,   47,   47,   47,   47,   47,

       19,   47,   21,    5,   47,   14,   47,   47,   47,    6,
       47,   47,   47,   30,   16,   47,   47,   17,   47,   47,
       47,   47,   47,   47,    2,   47,   47,   47,   47,   47,
       47,   47,   47,   47,   47,   20,   13,   18,   47,   15,
       22,   47,   47,   29,    1,    0
    } ;

static const YY_CHAR yy_ec[256] =
    {   0,
        1,    1,    1,    1,    1,    1,    1,    1,    2,    3,
        1,    1,    2,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    2,    1,    1,    1,    1,    4,    1,    1,    5,
        6,    7,    8,    9,   10,   11,   12,   13,   13,   13,
       13,   13,   13,   13,   13,   13,   14,   15,   16,
       17,   18,    1,    1,   19,   19,   19,   19,   19,   19,
       19,   19,   19,   19,   19,   19,   19,   19,   19,   19,
       19,   19,   19,   19,   19,   19,   19,   19,   19,   19,
       20,    1,   21,    1,   22,    1,   23,   24,   25,   26,

       27,   28,   29,   30,   31,   19,   19,   32,   33,   34,
       35,   36,   19,   37,   38,   39,   40,   41,   19,   19,
       42,   19,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,

        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1
    } ;
```

```
static const YY_CHAR yy_meta[43] =
    {   0,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        2,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    4,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
        1,    1
    } ;

static const flex_int16_t yy_base[150] =
    {   0,
        0,    0,  270,  271,   41,   43,  271,  271,  271,  271,
      271,  271,  271,  256,  271,   36,  251,  271,   33,  271,
      250,  246,  271,  271,    0,   30,   32,   26,   34,   30,
       33,   36,   44,   46,  243,   48,   49,   51,   50,   52,
      242,   81,  250,  249,   75,  250,  271,  271,  271,  271,
      238,    0,   58,   65,   67,   68,   75,   69,   70,   77,
       81,   83,   86,  237,   89,   95,   93,   87,   97,   98,
      245,  235,  101,  236,  102,  103,  105,  106,  107,  108,
      233,  111,  117,  110,  119,  123,  126,  128,  129,  130,
      131,  232,  134,  135,  138,  139,  140,  141,  132,  151,

      231,  149,  230,  229,  152,  228,  153,  154,  157,  227,
      165,  159,  167,  226,  225,  160,  161,  226,  171,  173,
      174,  175,  172,  184,  223,  185,  187,  189,  190,  191,
      193,  192,  195,  197,  198,  222,  221,  206,  199,  204,
      200,  196,  202,  109,   76,  271,  235,  236,  238
    } ;

static const flex_int16_t yy_def[150] =
    {   0,
      146,    1,  146,  146,  146,  146,  146,  146,  146,  146,
      146,  146,  146,  146,  140,  147,  146,  146,  146,  146,
      146,  148,  146,  146,  146,  149,  148,  148,  148,  148,
      148,  148,  148,  148,  148,  148,  148,  148,  148,  148,
      148,  146,  146,  146,  147,  147,  146,  146,  146,  146,
      148,  149,  148,  148,  148,  148,  148,  148,  148,  148,
      148,  148,  148,  148,  148,  148,  148,  148,  148,  148,
      146,  148,  148,  148,  148,  148,  148,  148,  148,  148,
      148,  148,  148,  148,  148,  148,  148,  148,  148,  148,
      148,  148,  148,  148,  148,  148,  148,  148,  148,  148,

      148,  148,  148,  148,  148,  148,  148,  148,  148,  148,
      148,  148,  148,  148,  148,  148,  148,  148,  148,  148,
      148,  148,  148,  148,  148,  148,  148,  148,  148,  148,
      148,  148,  148,  148,  148,  148,  148,  148,  148,  148,
      148,  148,  148,  148,  148,    0,  146,  146,  146
    } ;
```

```
static const flex_int16_t yy_nxt[314] =
    {   0,
        4,    5,    6,    7,    8,    9,   10,   11,   12,   13,
       14,   15,   16,   17,   18,   19,   20,   21,   22,   23,
       24,   25,   22,   22,   26,   27,   28,   29,   22,   30,
       31,   32,   33,   34,   35,   36,   37,   38,   39,   22,
       40,   41,   42,   42,   43,   42,   44,   52,   45,   48,
       49,   52,   53,   52,   52,   52,   58,   52,   54,   56,
       52,   60,   62,   57,   59,   52,   61,   52,   55,   52,
       52,   52,   52,   52,   63,   66,   69,   67,   70,   52,
       64,   68,   42,   42,   65,   44,   52,   45,   52,   52,
       52,   52,   74,   76,   72,   73,   52,   52,   52,   78,

       80,   82,   52,   81,   52,   75,   77,   52,   52,   84,
       52,   83,   85,   79,   52,   86,   52,   87,   52,   52,
       89,   88,   52,   52,   52,   94,   52,   52,   52,   52,
       52,   52,   52,   90,   91,   92,   93,   99,   52,   95,
       52,  100,   96,   97,   52,   98,  101,   52,  103,   52,
       52,   52,  102,   52,   52,   52,   52,  104,  108,   52,
       52,   52,  107,  106,   52,  105,  110,  105,  115,  112,
       52,  111,   52,   52,   52,   52,  120,  114,   52,  113,
       52,  116,  123,  119,   52,  118,   52,  117,   52,  122,
       52,  124,   52,   52,   52,   52,   52,  121,  128,  126,

      129,  130,  131,  125,  127,   52,   52,  133,   52,  134,
       52,   52,   52,   52,   52,  135,   52,   52,   52,   52,
       52,   52,  132,   52,  145,   52,  136,   52,  137,  138,
      144,  139,  140,  141,  142,  143,   46,   46,   51,   51,
       52,   52,   52,   52,   52,   52,   52,   52,   52,   52,
       52,   52,   52,   52,   52,   52,   52,   71,   52,   52,
      146,   71,   43,   52,   52,   52,   50,   47,   43,  146,
        3,  146,  146,  146,  146,  146,  146,  146,  146,  146,
      146,  146,  146,  146,  146,  146,  146,  146,  146,  146,
      146,  146,  146,  146,  146,  146,  146,  146,  146,  146,

      146,  146,  146,  146,  146,  146,  146,  146,  146,  146,
      146,  146,  146
    } ;
```



Implementación

El archivo y.tab.h que se incluye en el analizador léxico fue generado por yacc y contiene la información recolectada por yacc sobre los tipos de los atributos (declaración %union) y la enumeración de los terminales. Es, por tanto, necesario que la compilación con yacc preceda a la compilación con flex. La información en y.tab.h es usada por el analizador léxico para “sincronizarse” con el analizador sintáctico.

En la zona de declaraciones de expresiones regulares se escriben las expresiones para construir los identificadores, números y cadenas. Además se declaran los tokens que debe regresar el analizador, que además de los ya mencionados también incluyen las palabras reservadas del lenguaje que se procesa; se ignoran los espacios en blanco y los saltos de línea. Para los identificadores se utilizan las reglas del lenguaje C, por lo que así se define su construcción. Para las cadenas se permite ingresar letras y números.

Forma de ejecutar el programa

Para realizar el análisis léxico se requiere tener una función principal para leer el archivo de entrada. Para su ejecución se debe poner la siguiente instrucción en la consola: **“./nombre_ programa entrada”.**