1. Las expresiones regulares LEX.

Lex es un lenguaje que sirve para asociar acciones a la presencia de palabras (secuencias de caracteres) leídas de un fichero de tipo texto; las formas de las palabras que se pretende detectar se especifican en LEX mediante una notación que es una ampliación de las expresiones regulares.

Las expresiones regulares, tal y como se estudian en la asignatura “TALF” (*teoría de autómatas y lenguajes formales*), están constituidas por símbolos de un alfabeto Σ.

La precedencia de los operadores es la definida por la siguiente jerarquía, relacionada de mayor a menor precedencia:

1. Operaciones entre paréntesis.
2. Operador estrella.
3. Operador concatenación.

Entre los símbolos que aparecen en una expresión regular cabe distinguir los caracteres y los metacaracteres. Los primeros son los símbolos que pertenecen al alfabeto sobre el que está definida la expresión regular y los metacaracteres son los símbolos que no pertenecen a ese alfabeto: los operadores y los paréntesis.

En una especificación LEX se incluyen expresiones regulares, pero escritas con una notación que es una ampliación de éstas. Tiene como principales objetivos:

- Hacer más cómoda y breve la escritura de las expresiones regulares.
- Distinguir de manera precisa los caracteres del alfabeto y los metacaracteres empleados en la escritura de las expresiones regulares.

<u>El carácter de final de línea</u>

Un fichero de tipo texto está organizado en líneas y, por ello, se tiene un carácter especial que marca el final de cada una de las líneas que constituyen el fichero.

Si se mira el contenido de nuestro fichero como una secuencia ininterrumpida de caracteres de un cierto alfabeto, el carácter especial de final de línea ha de considerarse como un carácter más de la secuencia, es decir, se trata de un carácter del alfabeto al que pertenecen los caracteres grabados en el fichero.

El significado del espacio en blanco y del tabulador (horizontal)

En una especificación LEX las expresiones regulares se escriben siguiendo ciertas normas. Una de ellas indica que una expresión regular es una secuencia de caracteres cuya terminación está indicada mediante un espacio en blanco. En consecuencia el espacio en blanco también es un metacarácter que marca el final de la expresión.

Subexpresiones con nombre

Para hacer más cómoda o más legible la escritura de las expresiones regulares LEX se puede asociar un nombre a una expresión regular y, después, usar ese nombre como subexpresión de una expresión regular más compleja.

Para utilizar como componente de una expresión regular el nombre de otra expresión definida previamente basta con incorporar ese nombre delimitado por llaves en el lugar en el que proceda colocar la subexpresión.

El programa generado por Lex

Análisis de secuencias de caracteres

El problema que se pretende resolver con la ayuda de esta herramienta consiste en encontrar en un texto grabado en un fichero secuencias de caracteres que se ajusten a unas determinadas formas (a unos determinados modelos o patrones), y una vez encontrada una secuencia que se ajusta a un patrón proceder a la realización de unas operaciones que se tengan asociadas a ese patrón.

En la práctica habitual suelen tenerse varios patrones, cada uno de ellos con sus correspondientes operaciones (tareas) asociadas.

Las acciones se describen mediante código escrito en el lenguaje C debido a que:

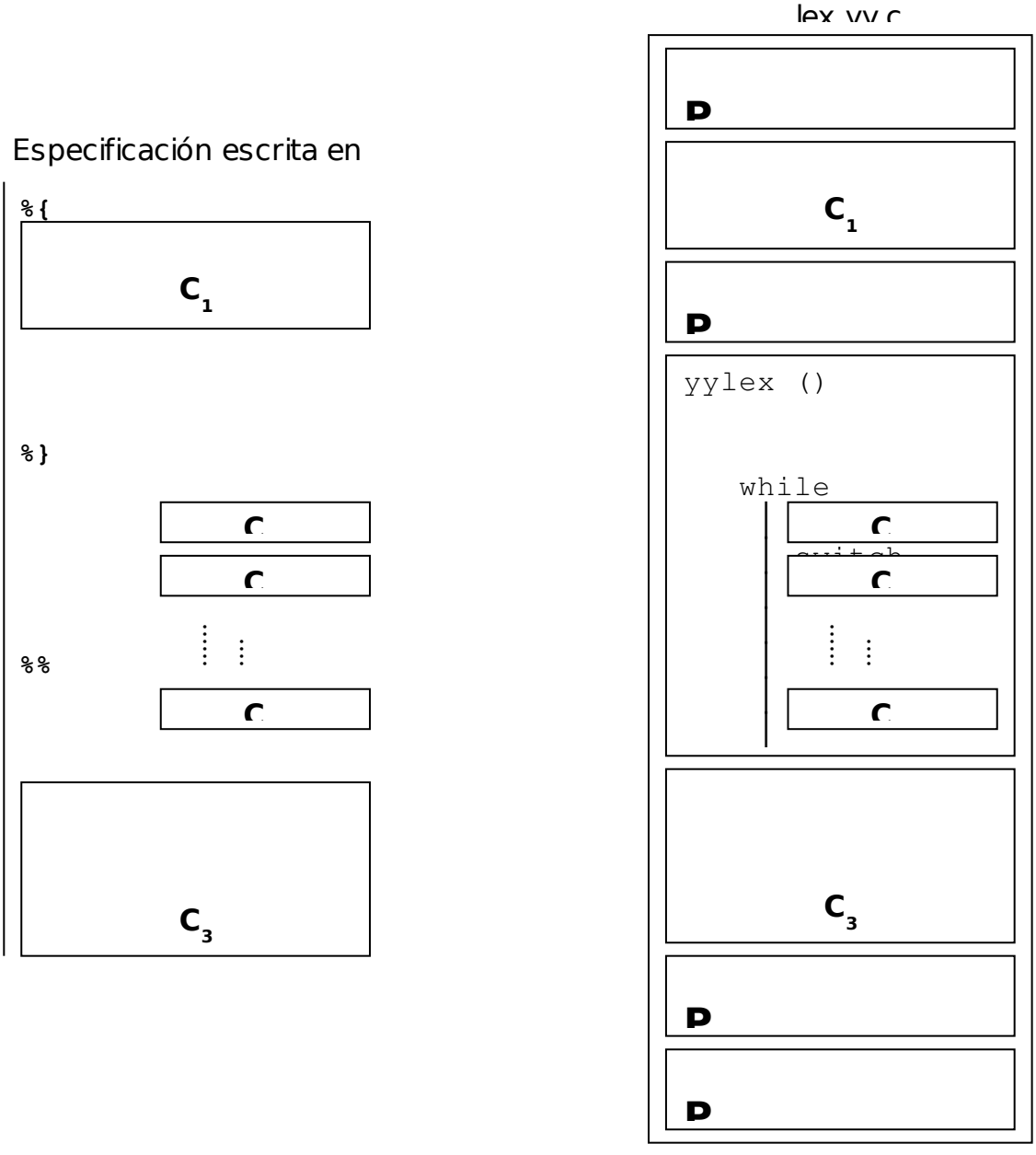
- El traductor LEX produce un analizador que es un programa escrito en C.
- El código de las acciones de la especificación de entrada se copia de manera literal a la salida, esto es, queda incorporado al programa escrito en C.

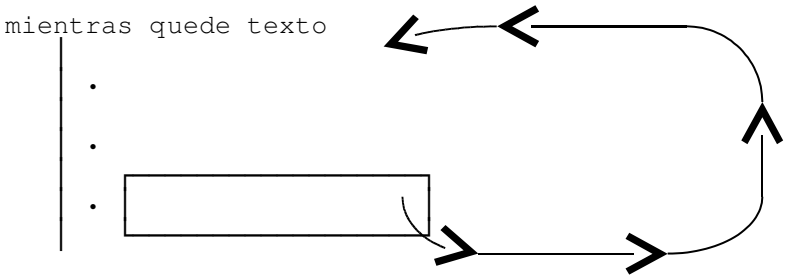
La salida producida por LEX es un programa cuya parte principal la constituye una función (yylex), que realiza el análisis de un texto según los patrones indicados en la especificación de la entrada. El algoritmo definido por la función está representado en el siguiente ciclo:

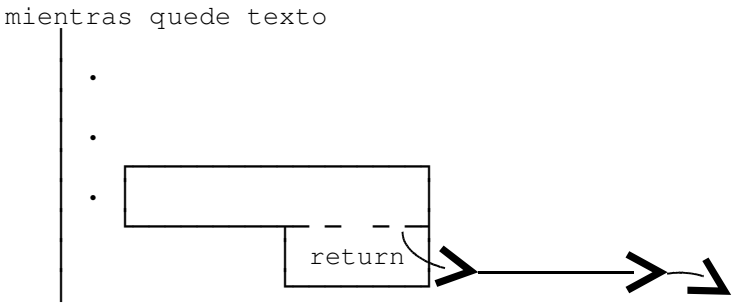
While (quede texto por analizar)
{
• acoplar un patrón a partir del punto actual de la entrada
• avanzar en la entrada hasta sobrepasar la secuencia acoplada
• ejecutar la acción asociada al patrón acoplado
}

Como se aprecia, cada vez que se llama a la función yylex, se trata de un ciclo repetido reiteradamente hasta que el fichero de entrada queda analizado por completo. Esto puede evitarse mediante sentencias return.

Los siguientes esquemas ilustran las dos maneras de funcionamiento; ambas son factibles y útiles: el uso de una u otra depende del problema que se quiera resolver con el programa generado por Lex.







2. Forma de una especificación LEX

Esquema general

La especificación Lex está compuesta por 3 secciones, tal y como se muestra en el esquema. La raya vertical de la izquierda representa el comienzo de las líneas del fichero texto de entrada, es decir la posición de la primera columna de cada línea.

Sección de reglas

Está formada por una secuencia de pares asociados: patrón, acción; el patrón es una expresión regular y la acción es un bloque de código escrito en C.

Cualquiera de las 3 secciones de una especificación Lex puede estar ausente (aunque la sección de reglas es la fundamental, y su ausencia hace que la especificación resulte inútil). El separador entre las secciones de definiciones y de reglas ha de estar presente, aunque ambas estén ausentes. El separador entre las secciones de reglas y de rutinas puede suprimirse cuando no hay sección de rutinas. Dentro de la sección de definiciones, cualquiera de sus 2 partes puede estar presente o no. Todos los separadores han de ponerse a partir de la primera columna.

Sección de rutinas

En esta sección se incluye código escrito en C que se traslada literalmente al fichero generado; lo que suele ponerse es el código de las funciones a las que se ha hecho referencia desde el código de las acciones de la sección de reglas. Aquí no hay que poner cuidado en la colocación del código en las líneas y columnas, se trasladará manteniendo la colocación y será un texto de entrada para el compilador de C.

El criterio de selección del patrón es:

1. A partir del carácter actual se intentan aplicar de manera simultánea todos los patrones de la especificación, habrá patrones que se puedan acoplar a la parte actual de la entrada, y otros que no se puedan.

2. De todos los patrones que se acoplan a la parte actual de la entrada, se selecciona el que se acopla a la secuencia de caracteres de mayor longitud (a partir del carácter actual).
3. Si ocurre que varios patrones se acoplan sobre la misma secuencia de longitud máxima, se selecciona de entre ellos el que está situado antes en la relación de patrones de la sección de reglas de la especificación.

Según este criterio, ocurre que, en general, no es indiferente el orden de colocación de los patrones en la sección de reglas.

Almacenamiento de la secuencia acoplada

El código generado por LEX a partir de la especificación de entrada realiza la tarea de dejar anotada la secuencia de caracteres a la que se acopla el patrón seleccionado.

Para este almacenamiento se dispone de dos variables globales cuyos nombres son:

yytext contiene la secuencia de caracteres acoplada
yyleng contiene la longitud de la secuencia acoplada

Como ocurre con el almacenamiento de las cadenas en el lenguaje C, el final de la secuencia de caracteres anotada se indica mediante el carácter especial de final de cadena.

El contenido de las variables yytext e yyleng cambia cada vez que se acopla (reconoce) una nueva secuencia definida por un determinado patrón; por ello, si la secuencia actual se necesita para alguna operación posterior, deberá trasladarse a otro lugar antes de que se acople un nuevo patrón.

3. Generación de analizadores léxicos

Justificación

En definitiva, un analizador léxico para un lenguaje L analiza la forma de los componentes básicos de un programa escrito en L. Esas formas, en la mayoría de los casos, pueden describirse mediante expresiones regulares. Por ello, la forma de todas las piezas sintácticas existentes en la definición lexicográfica del lenguaje puede constituir la secuencia de patrones de una especificación LEX. Se incluye un patrón por cada pieza sintáctica que deba reconocer el analizador lexicográfico.

La acción correspondiente al acoplamiento de un patrón a una cierta secuencia de caracteres (en este caso se tratará de un lexema) será la que corresponde a un analizador lexicográfico: devolver a la rutina llamante una representación de la pieza sintáctica encontrada en el texto de entrada.

Así pues, la función yylex generada a partir de una especificación debe de realizar la tarea de un analizador léxico:

1. Cada vez que se llame a yylex deberá avanzar en la entrada hasta encontrar la siguiente pieza sintáctica (siguiente token).

2. Las llamadas a yylex se realizan desde el analizador sintáctico.

Definición de los valores asociados a las piezas

La función generada yylex es de tipo entero, es decir, devuelve un valor de tipo entero. Por ello, los valores asociados a las piezas sintácticas han de ser números enteros. En principio, se puede elegir cualquier valor para la asociación. Es habitual, para favorecer la legibilidad, que se den nombres a los valores numéricos asociados a las piezas sintácticas; estos nombres se asignan mediante cláusulas #define incorporadas en la sección de definiciones.

Comunicación del lexema de una pieza sintáctica

El analizador léxico devolverá atributos en las variables globales antes mencionadas y dado que estas variables son globales, sus contenidos podrán consultarse desde el analizador sintáctico.

EOF (Final de fichero)

Cuando el analizador léxico (la función yylex), en el intento de encontrar el siguiente token se encuentra con el final del fichero, devuelve el valor cero.

Por esta razón, el número cero no puede emplearse para representar una pieza sintáctica, ni para indicar la presencia de un error.

Resumen

La comunicación entre los analizadores léxico y sintáctico se realizará de la siguiente manera.

1. El analizador léxico generado por LEX es la función “yylex”.
2. El analizador sintáctico produce sucesivas llamadas a “yylex”.
3. La función “yylex” devuelve un valor numérico representativo de la pieza sintáctica detectada en el texto de entrada.
4. En las variables globales yytext e yyleng se pueden consultar (si fuera preciso desde el analizador sintáctico) el lexema de la pieza y la longitud de ese lexema (secuencia de caracteres acoplada y sobrepasada).