Elementos léxicos del lenguaje de programación C

Elementos léxicos de los lenguajes de programación (tokens)

- Palabras reservadas
- Identificadores
- Literales
- Operadores
- Delimitadores
- Comentarios

Apéndices

- Construcción de expresiones en C
- Precedencia y asociatividad de los operadores en C

Elementos léxicos de C

Token

Componente léxico de un lenguaje de programación

Palabra reservada

Palabra que tiene un significado concreto en el lenguaje de programación, sin necesidad de que se lo asignemos nosotros.

```
auto        break       case        char
const       continue    default     do
enum        extern      float       for
goto        if          int         long
else        return      short       signed
sizeof      static      struct      double
register    switch      typedef     union
unsigned    void        volatile    while
```

Identificador

Palabra que podemos utilizar para denominar algo en el lenguaje.

Identificadores en C

- El primer símbolo del identificador será un carácter alfabético (a, …, z, A, …, Z, '_'). Después de ese primer carácter podremos poner caracteres alfanuméricos (a, …, z) y (0, 1, …, 9) y el guión de subrayado '_'.
- Las mayúsculas y las minúsculas se consideran diferentes.
- El guión de subrayado se interpreta como una letra más.
- Los identificadores no pueden coincidir con las palabras reservadas.

Ejemplos válidos
a, pepe, r456, tu_re_da, AnTeNa, antena

Ejemplos no válidos
345abc, mi variable, Nombre.Largo, cañada, camión

Literal

Especificación de un valor concreto de un tipo de dato.

Números

3 3.1416 0.31416e1 0.31416e+1 .31416e1f

Sufijos:	u|U	(unsigned)
	l|L	(long)
	f|F	(float)
Prefijos:	0	(octal)
	0x	(hexadecimal)

Caracteres

‘a’ ‘b’ ‘c’ ... ‘0’ ‘1’ ‘2’ ...

Secuencias de escape

\a	0x07	BEL	Sonido de alerta
\b	0x08	BS	Retroceso (Backspace)
\f	0x0C	FF	Salto de página (Form feed)
\n	0x0A	LF	Salto de línes (Line feed) = Nueva línea
\r	0x0D	CR	Retorno de carro
\t	0x09	HT	Tabulador horizontal
\v	0x0B	VT	Tabulador vertical
\\	0x5c	\	Barra invertida (Backslash)
\'	0x27	'	Apóstrofe = Comilla simple
\"	0x22	"	Comillas = Comilla doble
\?	0x3F	?	Signo de interrogación
\O			O = Cadena de dígitos octales
\xH			H = Cadena de dígitos hexadecimales
\XH			H = Cadena de dígitos hexadecimales

vg:	‘\0’	NULL	Carácter nulo

Cadenas de caracteres

“Cadena de caracteres en C”
“\”Mensaje entrecomillado\”\n”

Operador

Igual que en Matemáticas, realizan una acción específica:

- Suelen estar definidos en el núcleo del compilador
  (aunque también pueden estar definidos en bibliotecas externas)
- Suelen representarse con tokens formados por símbolos.
- Suelen utilizar notación infija.
- Pueden aplicarse a uno o varios operandos (argumentos).
- Suelen devolver un valor.

Delimitador

Símbolos utilizados como separadores de las distintas construcciones de un lenguaje de programación (esto es, los signos de puntuación de un lenguaje de programación.

```
[ ] ( ) { } ,   ;   :   ...  *   =   #
```

- **Corchetes:**
  Se utilizan para especificar los subíndices de los arrays.
- **Paréntesis:**
  Agrupan expresiones e indican los parámetros de las funciones.
- **Llaves:**
  Delimitan sentencias compuestas (bloques de código).
- **Comas:**
  Separan los elementos de una lista.
- **Punto y coma:**
  Indica el final de una sentencia.
- **Dos puntos:**
  Etiquetas.
- **Elipsis (...):**
  Funciones con un número variable de argumentos.
- **Asteriscos (*):**
  Punteros
- **Inicializador (=):**
  Separa la declaración de la inicialización de variables.
- **Almohadilla (#):**
  Directivas del preprocesador.

Comentario

Aclaración que el programador incluye en el texto del programa para mejorar su inteligibilidad.

Comentarios en C:

- Cualquier secuencia de caracteres que comience con `/*`.
- El comentario termina cuando se encuentra el par `*/`.
- No se pueden introducir comentarios dentro de otros

ANSI C también permite comentarios en una línea (al estilo de C++): Estos comentarios comienzan con dos barras // y terminan al final de la línea.

Apéndices

Construcción de expresiones en C

Operador de asignación

=

Operadores de comparación

== != >= <= > <

Operadores aritméticos

+ cast-expression
- cast-expression
add-expression + multiplicative-expression
add-expression - multiplicative-expression
multiplicative-expr * cast-expr
multiplicative-expr / cast-expr
multiplicative-expr % cast-expr

postfix-expression ++	(postincremento)
++ unary-expression	(preincremento)
postfix-expression --	(postdecremento)
-- unary-expression	(predecremento)

Operadores booleanos

logical-AND-expr && inclusive-OR-expression
logical-OR-expr || logical-AND-expression
! cast-expression

Operadores a nivel de bits

AND-expression & equality-expression
exclusive-OR-expr ^ AND-expression
inclusive-OR-expr | exclusive-OR-expression
~cast-expression
shift-expression << additive-expression
shift-expression >> additive-expression

Operadores compuestos de asignación

```
*=   /=   %=   +=   -=   <<=   >>=   &=   ^=   |=
```

Operador condicional

logical-OR-expr ? expr : conditional-expr

Operadores para el manejo de punteros

& cast-expression
* cast-expression

Operadores postfijos

postfix-expression(<arg-expression-list>)

array declaration [constant-expression]
compound statement { statement list }

postfix-expression . identifier
postfix-expression -> identifier

El operador sizeof

sizeof unary-expression
sizeof (type-name)

Precedencia y asociatividad de los operadores en C

Operador	Significado	Asociatividad
()	Llamada a una función	**De izquierda a derecha**
[]	Acceso a los elementos de un array	
.	Miembro de una estructua o unión	**De derecha a izquierda**
->	Miembro de una estructura o unión (punteros)	
sizeof	Tamaño de una variable / tipo	
++	Postincremento (v++)	
--	Postdecremento (v--)	
++	Preincremento (++v)	
--	Predecremento (--v)	
~	Complemento bit a bit	
!	Negación lógica	
-	Menos unario	
+	Más unario	
&	Referencia (Dirección de…)	
*	Indirección (Contenido de…)	
(tipo)	Casting (conversión de tipo)	
*	Multiplicación	**De izquierda a derecha**
/	División	
%	Resto	
+	Suma	
-	Resta	
<<	Desplazamiento de bits a la izquierda	
>>	Desplazamiento de bits a la derecha	
<	Menor que…	
<=	Menor o igual que…	
>	Mayor que…	
>=	Mayor o igual que…	
==	Igual que…	
!=	Distinto de…	
&	Operador AND (bit a bit)	
^	Operador XOR (bit a bit)	
|	Operador OR (bit a bit)	
&&	Operador AND (lógico)	
||	Operador OR (lógico)	
? :	Operador condicional	**De derecha a izquierda**
=	Operador de asignación **+= -= *= /= %= &= ^= |= >>= <<=**	