

FECHA 19, 2022

ANÁLISIS LÉXICO COMPILADORES



UNIVERSIDAD
Panamericana

Aspectos del análisis lexico



SEPARAR EL ANÁLISIS LEXICO DEL ANÁLISIS sintáctico (pe: eliminación de comentarios y espacios en blanco)

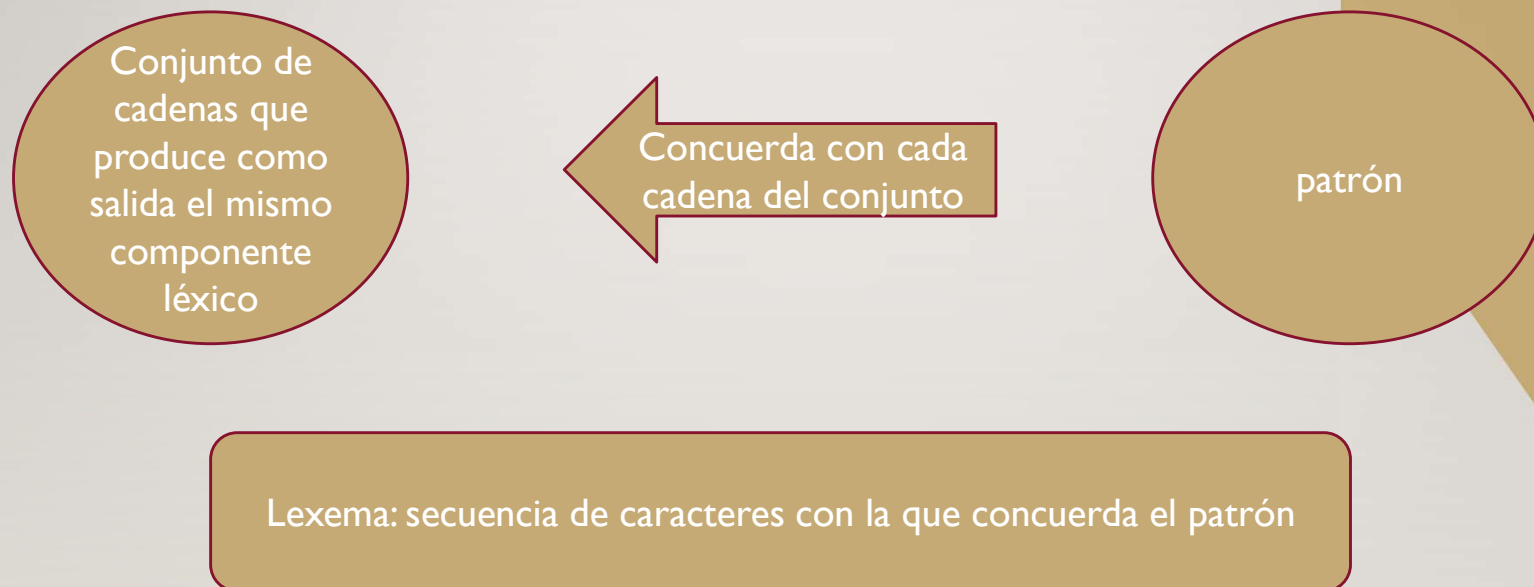


Gran parte del tiempo se requiere para leer el programa Fuente y dividirlo en components lexicos.



Lectura de símbolos especiales

Componentes léxicos, patrones y lexemas



Componentes léxicos, patrones y lexemas

const pi = 3.1416;

La subcadena pi es un lexema para el componente léxico “identificador”

Componente léxico	Lexemas de ejemplo	Descripción del patrón
const	const	const
if	if	if
relación	<, <=, >, >=, !=, ==	< o <= o > o >= o != o ==
id	pato, a, a2,	Letra seguida de letras y dígitos
num	0, 8.99, 6.0E23	Cualquier constante numérica
literal	“Hola mundo”	Cualquier carácter entre “ y ”, excepto “

Componentes léxicos, patrones y lexemas

Los componentes léxicos se tratan como terminales de la gramática del lenguaje fuente

Componentes léxicos



- Palabras clave
- Operadores
- Identificadores
- Constantes
- Cadenas literales
- Signos de puntuación (paréntesis, coma, punto y coma)

Componentes léxicos, patrones y lexemas

- Lenguajes modernos se tiende a la entrada independiente del formato.

PLI, FORTRAN necesitaban una entrada en columna, posición fija.

ATRIBUTOS DE LOS COMPONENTES LÉXICOS

$$E = a + b ^ 2$$

Lectura:

Id, puntero a la tabla de símbolos para E

Op_asignacion

Id, puntero a la tabla de símbolos para a

op_suma

Id puntero a la tabla de símbolos para b

Op_exponent

Num, valor entero 2

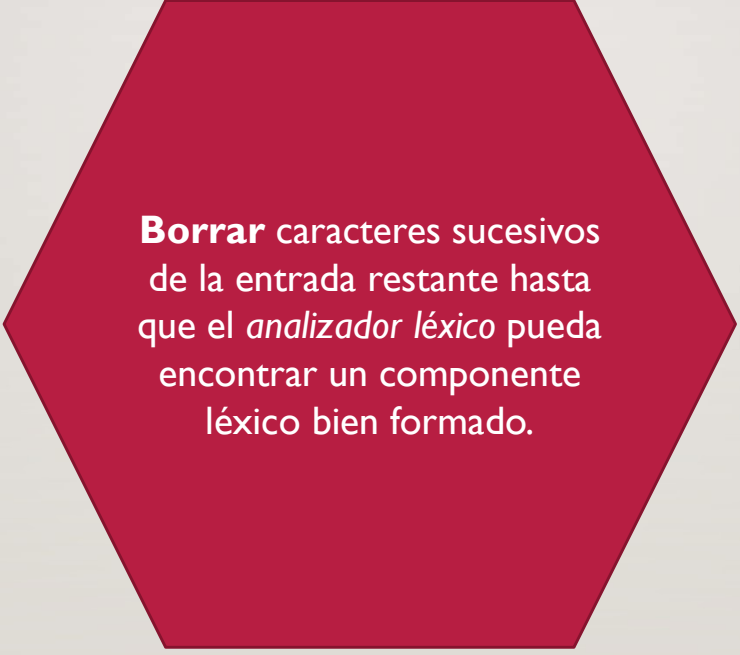
ERRORES LÉXICOS

fi (a < 100)...

- Palabra reservada mal escrita o identificador?
- Imagine una situación en la que el analizador léxico no pueda continuar porque ninguno de los patrones concuerda con un prefijo de la entrada restante....



ERRORES LÉXICOS



Borrar caracteres sucesivos de la entrada restante hasta que el *analizador léxico* pueda encontrar un componente léxico bien formado.

ERRORES LÉXICOS

Otras opciones:

- ❖ Borrar un carácter extraño
- ❖ Insertar un carácter que falta
- ❖ Reemplazar un carácter incorrecto por uno correcto
- ❖ Intercambiar caracteres adyacentes

Tratar de reparar la entrada

MANEJO DE BUFFERS DE ENTRADA

- Considerar que el analizador léxico es la parte mas lenta de la compilación por la lectura del archivo.
- Según el Lenguaje de Programación para realizar el compilador se deberán analizar las estrategias de lectura que éste ofrezca, para reducir el número de operaciones necesarias para procesar los elementos de entrada.

DIAGRAMAS DE TRANSICIONES

- Es un diagrama de flujo estilizado que representan las acciones que tienen lugar cuando el analizador léxico es llamado por el analizador sintáctico para obtener el siguiente componente léxico.
- Cambia la posición en el diagrama según se leen los caracteres.
- Las posiciones en un diagrama de transición de llaman *estados* y se representan con un círculo.
- Los estados se conectan mediante flechas llamadas *aristas*.
- Las aristas que salen de un estado tienen etiquetas que indican los caracteres de entrada que pueden aparecer después de haber llegado a dicho estado.

DIAGRAMAS DE TRANSICIONES

- Son diagramas **deterministas**, que ningún símbolo puede concordar con las etiquetas de dos aristas que salgan de un estado.
- Un estado es etiquetado como el estado de inicio.
- Al entrar en un estado se lee el siguiente carácter de entrada. Si existe una arista del estado actual cuya etiqueta concuerde con el carácter de entrada, entonces se mueve de estado. De otra forma, se indica que hubo un fallo.

DIAGRAMAS DE TRANSICIONES

- El círculo doble indica que es un estado de aceptación, en el cual se ha encontrado un componente léxico.

Se usa un *
para indicar los
estados en los
que se debe
llevar a cabo
retroceso en la
entrada

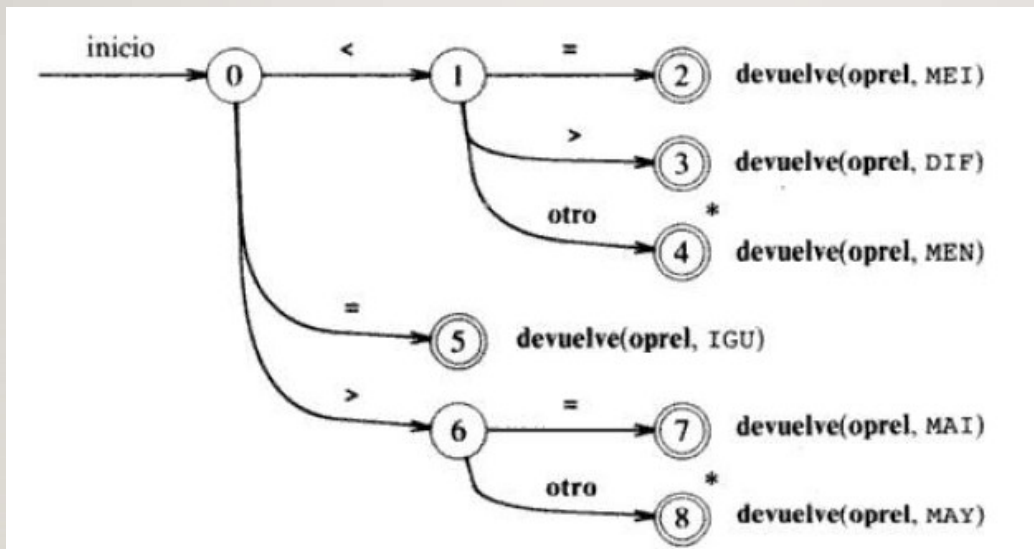
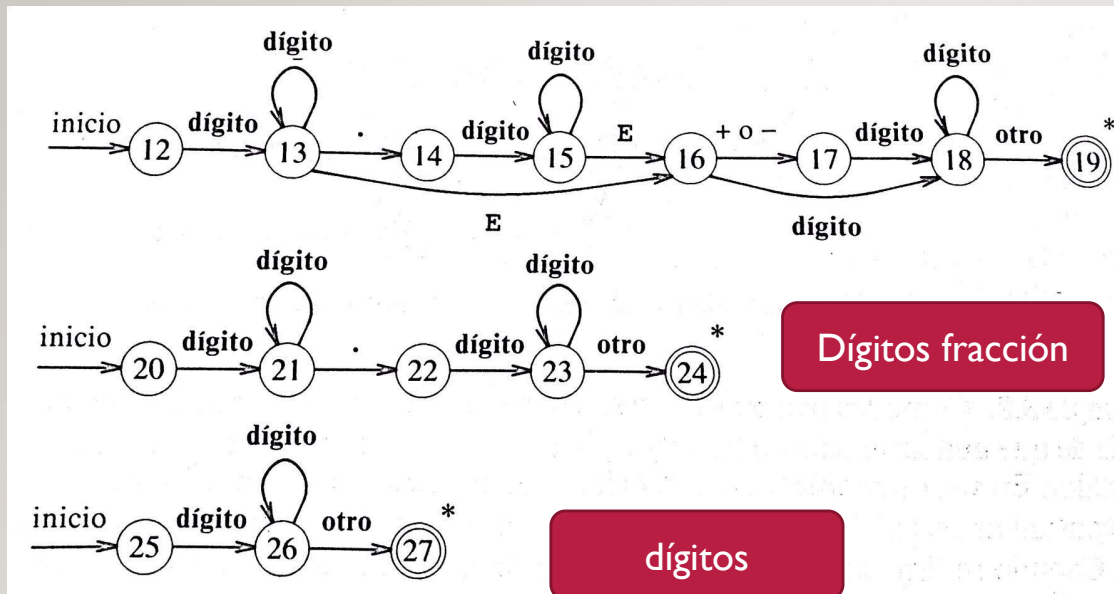


Diagrama de transiciones
para operadores
relacionales

DIAGRAMAS DE TRANSICIONES



Dígitos fracción? exponente

Dígitos fracción

dígitos

Reconocedor de números

num -> **dígito**⁺(**.dígito**⁺)?(**E(+ | -)?dígito**⁺)?

Entrada
12.3E4

El lexema de un determinado componente léxico debe ser el mas grande posible

DIAGRAMAS DE TRANSICIONES

- Los estados de aceptación introducen el lexema en una tabla de números y devuelve un apuntador a la entrada creada.
- El analizador léxico devuelve el componente léxico **num** con el apuntador como valor léxico.
- `l.>x` fracasa en los estados 14 y 22.
 - Puede regresar el `l`
 - Puede regresar el `l.0` para completar con `l.0>x`

Manejo de errores

IMPLANTACIÓN DE UN DIAGRAMA DE TRANSICIONES

- Programa que busque los componentes léxicos especificados por los diagramas.
- Su tamaño es proporcional al número de estados y de aristas de los diagramas.
- A cada estado le corresponde un segmento de código, si hay aristas que salen de un estado, entonces lee un nuevo carácter y selecciona una arista para seguir si es posible.
- Si no hay tal arista, y el estado en curso de ejecución no indica que se ha encontrado un componente léxico entonces se llama a una rutina de fallo y regresar a probar otro diagrama de transiciones.
- Si no hay mas diagramas de transiciones para probar, se llama a una rutina de recuperación de errores.

```

complex sigte_complex()
{
    while(1) {
        switch (estado) {
            case 0:      c = sigtecar();
                        /* c es el carácter de preanálisis */
                        if (c==blanco || c==tab || c==línea_nueva) {
                            estado = 0;
                            inicio_lexema++;
                            /* se avanza el inicio del lexema */
                        }
                        else if (c == '<') estado = 1;
                        else if (c == '=') estado = 5;
                        else if (c == '>') estado = 6;
                        else estado = fallo();
                        break;
                        .../* aquí van los casos del 1 al 8 */
            case 9:      c = sigtecar();
                        if (isletter(c)) estado = 10;
                        else estado = fallo();
                        break;
            case 10:     c = sigtecar();
                        if (isletter(c)) estado = 10;
                        else if (isdigit(c)) estado = 10;
                        else estado = 11;
                        break;
            case 11:     regresa(1); instala_id;
                        return ( obtén_complex() );
                        .../* aquí van los casos del 12 al 24 */
            case 25:     c = sigtecar();
                        if (isdigit(c)) estado = 26;
                        else estado = fallo();
                        break;
            case 26:     c = sigtecar();
                        if (isdigit(c)) estado = 26;
                        else estado = 27;
                        break;
            case 27:     regresa(1); instala_núm();
                        return( NUM );
        }
    }
}

```

IMPLANTACIÓN DE UN DIAGRAMA DE TRANSICIONES

AUTÓMATAS FINITOS

Un *reconocedor* de un lenguaje es un programa que toma como entrada una cadena x y responde “sí” si x es una frase del programa y “no” si no lo es.

Una **expresión regular** → **reconocedor** construyendo un diagrama de transiciones generalizado llamado autómata finito.

AUTÓMATAS FINITOS, NO DETERMINISTAS

En un estado se puede dar el caso de tener mas de una transición para el mismo símbolo de entrada.

AUTÓMATA FINITO NO DETERMINISTA

Un autómata finito no determinista (AFN) es un modelo matemático formado por:

1. Un conjunto de *estados* S
2. Un conjunto de símbolos de entrada Σ (*alfabeto de símbolos de entrada*)
3. Una función de transición que transforma pares estado-símbolo en conjuntos de estados.
4. Un estado S^0 que se considera *estado de inicio* (o inicial)
5. Un conjunto de estados F considerados *estados de aceptación* (o finales)

Se puede representar mediante un grafo dirigido llamado *grafo de transiciones*, en el que los nodos y las aristas representan la función de transición.

AUTÓMATA FINITO NO DETERMINISTA

Lenguaje $(a|b)^* abb$

$S=\{0,1,2,3\}$

$\Sigma=\{a,b\}$

$S_0=\{0\}$

$F=\{3\}$

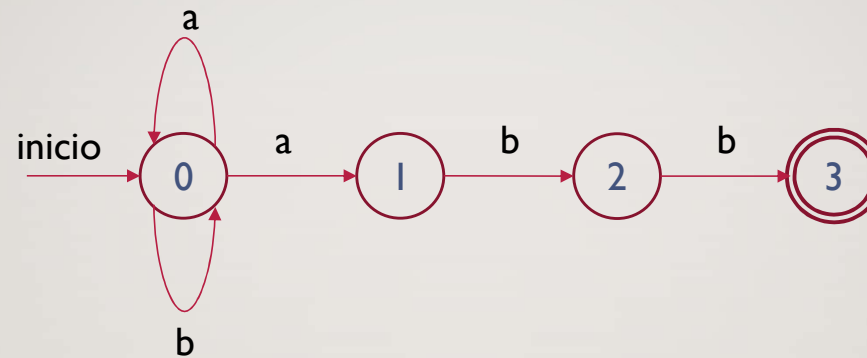


Tabla de transiciones

$S \backslash \Sigma$	a	b
0	$\{0,1\}$	$\{0\}$
1		$\{2\}$
2		$\{3\}$

AUTÓMATA FINITO NO DETERMINISTA CON TRANSICIONES ÉPSILON (AFND- ϵ)

Es un autómata finito no determinista en donde se permiten transiciones **que no contengan ningún símbolo de la entrada**. Es decir, se puede pasar de un estado a otro **sin consumir** ningún símbolo de la entrada.

AUTÓMATA FINITO NO DETERMINISTA

Lenguaje $aa^* \mid bb^*$

$S =$

$\Sigma =$

$S_0 =$

$F =$

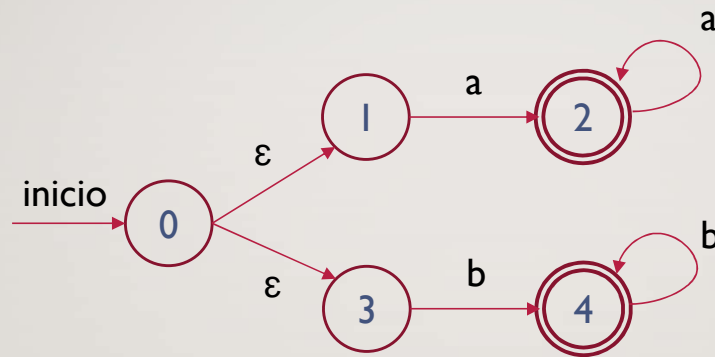


Tabla de transiciones

$S \backslash \Sigma$			

EJERCICIOS

1. Construya un AFND- ϵ para el lenguaje en $\Sigma = \{0,1\}$ cuyas cadenas empiezan con "0" o terminan en "1".
2. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma = \{a,b,c\}$. El conjunto de cadenas que inician en la sub-cadena "ac" o terminan en la sub-cadena "ab".
3. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma = \{0,1\}$. El conjunto de cadenas que inician en "01" o terminan en "0".
4. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma = \{a,b\}$. El conjunto de cadenas que inician en "b" o terminan en "bba".
5. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma = \{a,b,c\}$. El conjunto de cadenas que inician en la sub-cadena "ac" y terminan en la sub-cadena "ab".
6. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma = \{0,1\}$. El conjunto de cadenas que inician en "0" y terminan en "1".
7. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma = \{0,1\}$. El conjunto de cadenas que inician en "01" y terminan en "0".
8. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma = \{a,b\}$. El conjunto de cadenas que inician en "b" y terminan en "bba".
9. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma = \{a,b,c\}$. El conjunto de cadenas que inician en la sub-cadena "ac" o no terminan en la sub-cadena "ab".
10. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma = \{0,1\}$. El conjunto de cadenas que inician en "0" o no terminan en "1".
11. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma = \{0,1\}$. El conjunto de cadenas que inician en "01" o no terminan en "0".
12. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma = \{a,b\}$. El conjunto de cadenas que inician en "b" o no terminan en "bba".

EJERCICIOS

13. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{a,b,c\}$. El conjunto de cadenas que inician en la sub-cadena "ac" y no terminan en la sub-cadena "ab".
14. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{0,1\}$. El conjunto de cadenas que inician en "0" y no terminan en "1".
15. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{0,1\}$. El conjunto de cadenas que inician en "01" y no terminan en "0".
16. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{a,b\}$. El conjunto de cadenas que inician en "b" y no terminan en "bba".
17. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{a,b,c\}$. El conjunto de cadenas que no inician en la sub-cadena "ac" o no terminan en la sub-cadena "ab".
18. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{0,1\}$. El conjunto de cadenas que no inician en "0" o no terminan en "1".
19. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{0,1\}$. El conjunto de cadenas que no inician en "01" o no terminan en "0".
20. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{a,b\}$. El conjunto de cadenas que no inician en "b" o no terminan en "bba".
21. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{a,b,c\}$. El conjunto de cadenas que no inician en la sub-cadena "ac" y no terminan en la sub-cadena "ab".
22. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{0,1\}$. El conjunto de cadenas que no inician en "0" y no terminan en "1".
23. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{0,1\}$. El conjunto de cadenas que no inician en "01" y no terminan en "0".
24. Obtenga un AFND- ϵ dado el siguiente lenguaje definido en el alfabeto $\Sigma=\{a,b\}$. El conjunto de cadenas que no inician en "b" y no terminan en "bba".

REFERENCIAS ANEXAS AL LIBRO BASE.

- Dean K. (1995). "Teoría de Autómatas y Lenguajes Formales". Edit. Prentice Hall, España. "
- Hopcroft J. E., Ullman J.D. (2007). "Introducción a la teoría de autómatas, lenguajes y computación". 3ª ed. Edit. Pearson Educación, Madrid. "
- Linz P. (2001) "An Introduction to Formal Languages and Automata", 3rd Edition, J.A. Bartlett. "
- Martin J. (2004). "Lenguajes Formales y Teoría de la computación". 3ª ed. Edit. MacGraw-Hill Interamericana de México.