

# Analizador Léxico

**Desarrollador**

*Juan Cardona*



**UNIVERSIDAD  
DE ANTIOQUIA**  
1 8 0 3

Facultad de Ingeniería

Departamento de ingeniería de sistemas

Teoría de Lenguajes

2021-1

# Introducción

Un Analizador léxico es la primera fase de un compilador, su función es la de leer la cadena de caracteres de entrada e identificar cada componente léxico o lexema asignándole un token específico.

El método que se usó para construir el analizador léxico se realizó en los siguientes pasos:

Creación de la lista de componentes léxicos

AFD: Diagramas de burbujas para cada clase de lexema

AFD: Tablas de transiciones para cada clase de lexema

Tabla de lexemas y tokens

Desarrollo de la aplicación

# Desarrollo

## Lista de componentes léxicos

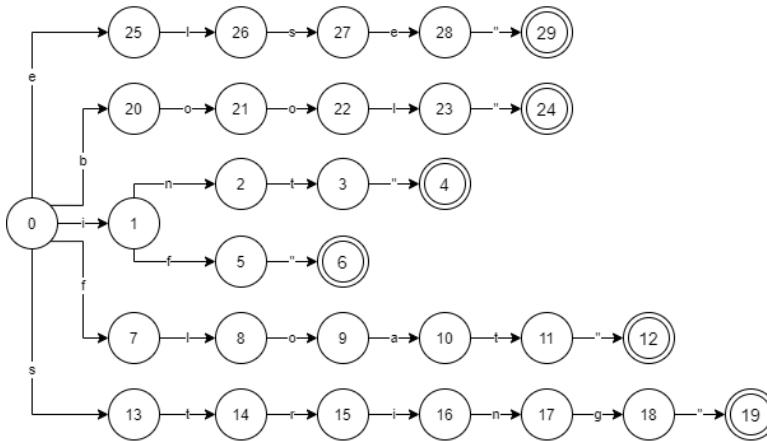
El analizador léxico acepta palabras reservadas, identificadores, constantes numéricas, constantes alfabéticas, operaciones aritméticas y separadores.

- Palabras reservadas (keyword):
  - int
  - float
  - string
  - bool
  - if
  - else
- Identificadores (identify): nombres de variables, funciones o clases creadas por el usuario. Constantes alfanuméricas, no inician con un número.
- Constantes numéricas (numConst): números enteros positivos y números flotantes.
  - D (cualquier caracter numérico o dígito)
  - ,
- Constantes alfabéticas (charConst): cadenas de caracteres
  - L (cualquier carácter alfabético, en minúscula o mayúscula, o letra)
- Operaciones aritméticas (operator):
  - +
  - -
  - \*
  - /
  - =
  - <
  - <=
  - <>
  - >
  - >=
  - ==
- Separadores (separator):
  - (
  - )
  - {
  - }
  - ;

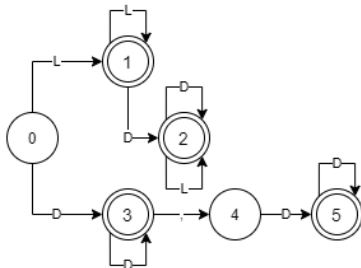
## AFD: Diagramas de burbujas para cada clase de lexema

Para los AFD se consideraron una lista de caracteres de ingreso partiendo de la lista de componentes léxicos y se seleccionaron, mediante los caracteres necesarios para asignarle a cada tipo de lexema, un token específico correspondiente en los AFD a los estados de aceptación.

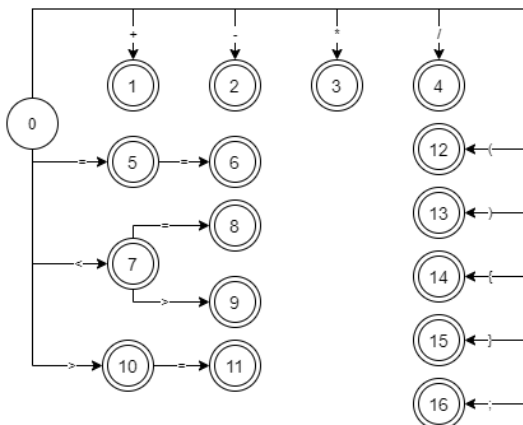
AFD1: keyword



AFD2: identify, numConst, charConst



AFD3: operator, separator



## AFD: Tablas de transiciones para cada clase de lexema

Partiendo de cada diagrama de burbujas se determinaron las siguientes tablas de transiciones y se le asignaron a cada estado de aceptación un token correspondiente al lexema que se identifica en las secuencias de símbolos de entrada.

ADF1: keyword

	e	s	l	b	o	g	r	n	t	f	a	d	u	i	
0	20	10		16						5				1	0
1								2		4					0
2									3						0
3															1
4															1
5			6												0
6					7										0
7											8				0
8									9						0
9															1
10									11						0
11								12							0
12														13	0
13								14							0
14						15									0
15															1
16					17										0
17					18										0
18			19												0
19															1
20			21												0
21		22													0
22	23														0
23															1

INT

IF

FLOAT

STRING

BOOL

ELSE

AFD2: identify, numConst, charConst

- D (cualquier caracter numérico o dígito)
- L (cualquier carácter alfabético, en minúscula o mayúscula, o letra)

	L	D	,	
0	1	3		0
1	1	2		1
2	2	2		1
3		3	4	1
4		5		0
5		5		1

CHARCONST  
IDENTIFY  
INTCONST  
FLOATCONST

ADF3: operator, separator

	+	-	*	/	=	<	>	(	)	{	}	;	
0	1	2	3	4	5	7	10	12	13	14	15	16	0
1													1
2													1
3													1
4													1
5					6								1
6													1
7					8		9						1
8													1
9													1
10					11								1
11													1
12													1
13													1
14													1
15													1
16													1

PLUS  
MINUS  
MULTIPLICATION  
DIVISION  
ASIGNATION  
BOOLEANCOMPARISON  
MINOR  
MINOREQUAL  
DIFFERENT  
HIGHER  
HIGHEREQUAL  
OPENPARENTHESIS  
CLOSEPARENTHESIS  
OPENBRACKETS  
CLOSEBRACKETS  
SEMICOLON

## Tabla de lexemas y tokens

De cada estado de aceptación obtenemos la lista de tokens para cada lexema que reconoce el analizador léxico, nótese que el espacio en blanco no es tomado en cuenta, su presencia es irrelevante.

Lexema	Token
esle	ELSE
bool	BOOL
int	INT
if	IF
float	FLOAT
string	STRING
$L^+$	CHARCONST
$L^+(L+D)^*$	IDENTIFY
$D^+$	INTCONST
$D^+, D^+$	FLOATCONST
+	PLUS
-	MINUS
*	MULTIPLICATION
/	DIVISION
=	ASIGNATION
==	BOOLCOPARISON
<	MINOR
<=	MINOREQUAL
<>	DIFFERENT
>	HIGHER
>=	HIGHEREQUAL
(	OPENPARENTHESIS
)	CLOSEPARENTHESIS
{	OPENBRACKETS
}	CLOSEBRACKETS
;	SEMICOLON

Estos lexemas  
se representan  
por una  
expresión  
regular

## **Desarrollo de la aplicación**

Para ver el código, la documentación y el archivo ejecutable con el analizador léxico visitar el siguiente repositorio: <https://github.com/ohmono/analizador-lexico>

## **Bibliografía**

- Flórez Rueda, R. (2010). Introducción a los compiladores. Medellín, Colombia: Universidad de Antioquia