



**UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA**

*La Universidad Católica de Loja*

# ANALIZADOR LÉXICO

**ESTUDIANTE:**

**ERIKA TATIANA VÁSQUEZ  
TAPIA.**

**DOCENTE:**

**JUAN CARLOS TORRES.**

**CARRERA:**

**SISTEMAS INFORMÁTICOS Y  
COMPUTACIÓN**

**COMPONENTE EDUCATIVO:**

**TEORIA DE AUTÓMATAS Y  
COMPILADORES.**

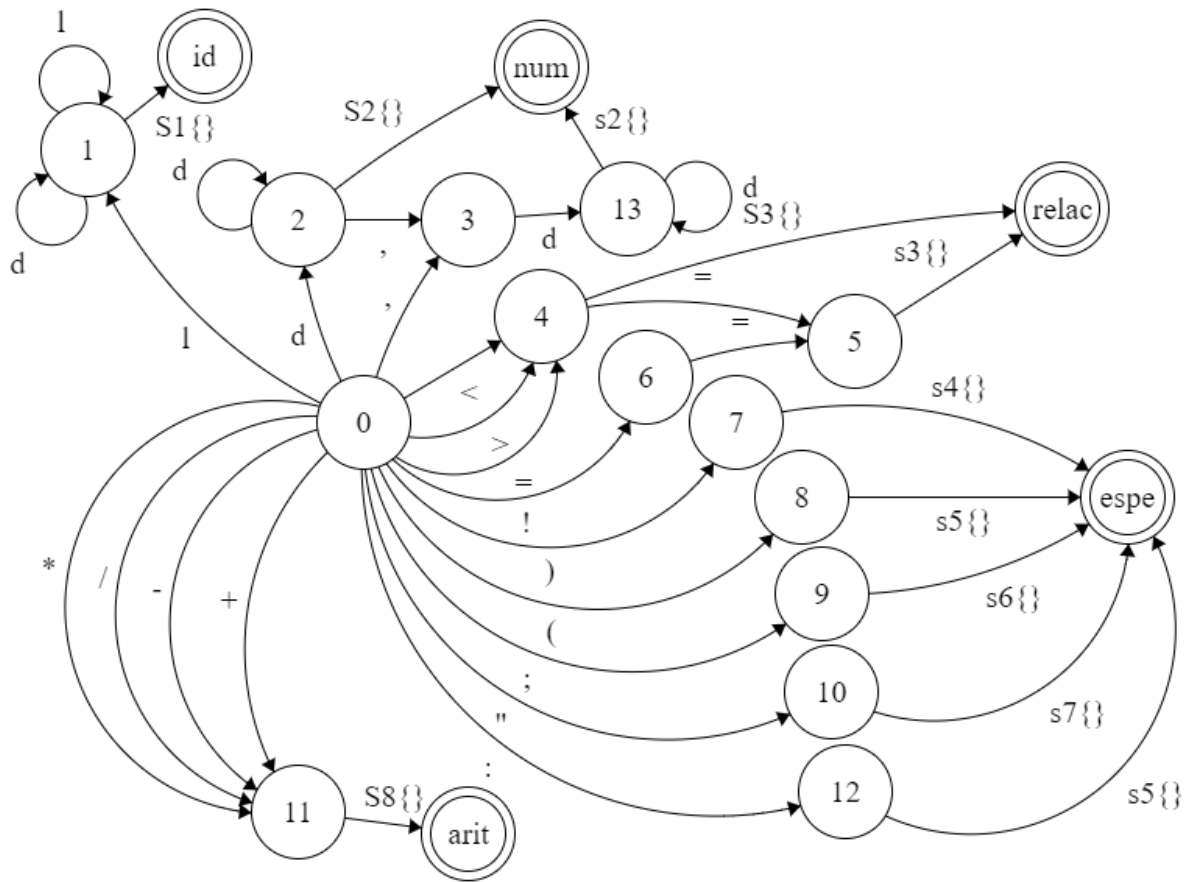
**FECHA:**

**20 NOVIEMBRE DE 2018**

### DISEÑO DEL AUTÓMATA.

El autómata desarrollado tiene un total de 12 estados, en el cual el estado inicial está definido por el número 0, del mismo que se desglosan todos los posibles casos que pueden suceder. en la Fig.1 se puede observar el diagrama del autómata y como se relaciona lógicamente a cada uno de los estados y todos los conjuntos de separadores que existen en cada uno de los estados finales.

Dando un total de 8 conjuntos de separadores, tanto para números, palabras, operadores aritméticos, operadores relacionales o caracteres especiales.



**Fig.1|** Diseño del Autómata, desarrollado en Finite State Machine Designer.

## DEFINICIÓN DEL LENGUAJE DE PROGRAMACIÓN

PALABRAS RESERVADAS		
Grupo	Palabra	Descripción
Inicio y Fin de programa	<b>start</b>	Inicio de programa.
	<b>finish</b>	Fin de programa.
Ingreso y salida de datos	<b>set</b>	Imprimir por pantalla.
	<b>get</b>	Ingresar por teclado.
Tipos de Variables	<b>int</b>	Variable Entera.
	<b>string</b>	Variable Cadena.
	<b>double</b>	Variable Decimal.
	<b>boolean</b>	Variable Booleana.
Operador Lógico	<b>and</b>	operador relacional AND.
	<b>or</b>	Operador relacional OR.
Estructuras de Control	<b>while</b>	Ciclo de repetición.
	<b>endwhile</b>	Fin del ciclo mientras.
	<b>if</b>	Condicional.
	<b>endif</b>	Fin de condicional.
	<b>else</b>	Caso contrario condicional.
CONJUNTO DE CARACTERES		
Operadores Relacionales	<b>&lt;</b>	Menor que.
	<b>&gt;</b>	Mayor que.
	<b>&lt;=</b>	Menor o igual que.
	<b>&gt;=</b>	Mayor o igual que.
	<b>==</b>	Igual que.
	<b>!=</b>	Diferente que.
Caracteres Especiales	<b>;</b>	Terminar una línea de código.
	<b>(</b>	Iniciar una expresión.
	<b>)</b>	Terminar una expresión.
	<b>=</b>	Asignar valor a las variables.
	<b>,</b>	Separa caracteres.
	<b>.</b>	Separa caracteres.
Operadores Aritméticos	<b>+</b>	Suma
	<b>-</b>	Menos
	<b>*</b>	Multiplicación
	<b>/</b>	División

## REGLAS DEL LENGUAJE

REGLAS	
La ejecución del programa comienza con la palabra <b>"start:"</b> y termina con <b>"finish;"</b> . Al final de cada expresión debe haber un carácter <b>","</b> .	<b>start:</b> <b>expresión;</b> <b>finish;</b>
Para la declaración de variables se debe seguir la siguiente estructura. Tipo de variable, nombre de la variable, signo de igualdad "=", valor de la variable y carácter especial ",".	<b>palabra reservada nombre = valor;</b>  int suma = 5;
Una variable nunca podrá iniciar con un numero o carácter especial.	variable suma; variable s1; variable _suma5; <b>ERROR</b>
La obtención de valores se hará asignando la en valor en una variable mediante la palabra reservada <b>get</b> , seguido del signo '(', mensaje y termina con ')';.	<b>variable = get ("mensaje de entrada");</b>  suma = get("Ingrese un valor");
La impresión de valor se realizará mediante la palabra <b>post</b> , seguido del signo '(', el mensaje que se desea presentar y termina con ')';. En caso de imprimir el valor de una variable se debe agregar el carácter "," después del carácter ");	<b>set("mensaje de salida");</b>  post("Hola"); post("El valor es:", suma);
Las estructuras de control y bucles, dentro de los caracteres "(expresión):", deberá haber una expresión lógica. El signo ":" significa la culminación de la expresión lógica. Para separar las expresiones que van dentro de una estructura de control se debe tabular correctamente.	<b>if (:</b> <b>expresión;</b> <b>else</b> <b>expresión;</b> <b>endif;</b> <b>while(:</b> <b>expresión;</b> <b>Endwhile;</b>

## CONJUNTO DE SEPARADORES

### Separadores de las palabras:

```
Spalabra [ ] = { " " , " ; " , " . " , " ( " , " = " , " > " , " < " , " ! " , " + " , " - " , " / " , " * " , " \ " " " ) " " } ;
```

### Separadores de los números:

```
SNumeros [ ] = { " " , " = " , " > " , " < " , " ! " , " + " , " - " , " / " , " * " , " ; " , " ) " " } ;
```

### Separadores de los caracteres <,>, <=, >=, =, ==, != :

```
SRelacional [ ] = { " " , " ( " , " \ " " " } ;
```

**Separadores del carácter ( :**

```
SRelacional2 [ ] = { " " , " \" " , " ( " } ;
```

**Separadores del carácter ) :**

```
SRelacional1 [ ] = { " " , ";" , ")" } ;
```

**Separadores del carácter ; y :**

```
SRelacional3 [ ] = { " " } ;
```

**Separadores del carácter “ :**

```
SRelacional4 [ ] = { " " , ")" , " , " , "+" , " \" " } ;
```

**Separadores Aritméticos:**

```
SAritmeticos [ ] = { " " , ";" , "+" , "=" } ;
```

## IMPLEMENTACION DEL AUTOMATA

El desarrollo del Autómata se lo realizó en el lenguaje C++. A continuación, en la Fig. 2 se muestra una parte del código implementado en la ejecución del programa. Mismo que nos muestra parte de lo que es el estado inicial 0,

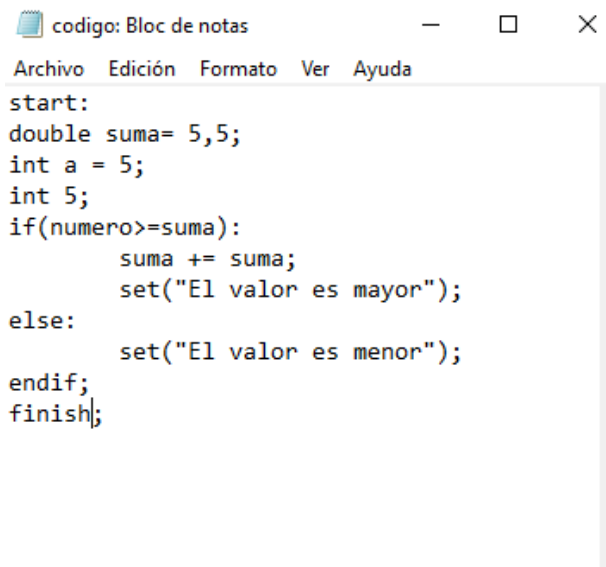
```
switch(estado) {  
    case 0:  
        // LETRA  
        if(isalpha(caracter)){  
            estado = 1;  
            cadena = cadena + caracter;  
            break;  
        }  
        // DIGITO  
        }else if (isdigit(caracter)){  
            estado = 2;  
            cadena = cadena + caracter;  
            break;  
        }  
        //COMA DIGITO  
        }else if(caracter == ','){  
            estado = 3;  
            cadena = cadena + caracter;  
            break;  
        }  
        //RELACIONALES  
        }else if(caracter == '<' || caracter == '>' || caracter == '='){  
            estado = 4;  
            cadena = cadena + caracter;  
            break;  
        }  
        //RELACIONALES  
        }else if(caracter == '!'){  
            estado = 5;  
            cadena = cadena + caracter;  
            break;  
        }  
        //ESPECIALESx  
        }else if(caracter == ')'){  
            estado = 7;
```

**Fig.2|** Código de la Implementación del Autómata, desarrollado en Dev-C++

## RESULTADOS OBTENIDOS

Una vez codificado en autómata, se presentan los siguientes resultados al analizar el siguiente documento de texto que se observa en la Fig. 3.

El resultado de la Fig.4 que nos indica el total de tokens correctos, la palabra y el tipo de dato que es cada uno de las pablas ingresadas. En cambio, Fig.5 tiene un código erróneo, que una vez ejecutado nos lanza los errores que se obtienen y la posición en la cual se encuentra guiándonos por los tokens que se obtiene en la Fig. 6



```
start:
double suma= 5,5;
int a = 5;
int 5;
if(numero>=suma):
    suma += suma;
    set("El valor es mayor");
else:
    set("El valor es menor");
endif;
finish;
```

**Fig.3|** Texto de Prueba Correcto.

ANALIZADOR LEXICO			
UTPL			
Estudiante: Erika Tatiana Vasquez Tapia			
NRO.	PALABRA	TOKEN	TIPO
- 1	start	palabra	Palabra Reservada
- 2	:	:	Caracter Especial
- 3	double	palabra	Palabra Reservada
- 4	suma	palabra	Identificador
- 5	=	=	Asignacion
- 6	5,5	numero	Valor Decimal
- 6	;	;	Caracter Especial
- 7	int	palabra	Palabra Reservada
- 8	a	palabra	Identificador
- 9	=	=	Asignacion
- 10	5	numero	Valor Entero
- 10	;	;	Caracter Especial
- 11	int	palabra	Palabra Reservada
- 12	5	numero	Valor Entero
- 12	;	;	Caracter Especial
- 13	if	palabra	Palabra Reservada
- 14	(	(	Caracter Especial
- 15	numero	palabra	Identificador
- 16	>=	>=	Operador Relacional
- 17	suma	palabra	Identificador
- 18	)	)	Caracter Especial

**Fig.4|** Resultado de la Ejecución del Analizador Léxico, Código Correcto.

```

start
double suma= 5,5fdf;
int a = 5;
int 5;
if(numero>=):
    suma += suma;
    set("El valor es mayor");
else:
    set("El valor es menor");
endif;
finish;

```

**Fig.5/** Texto de Prueba Incorrecto.

ANALIZADOR LEXICO			
UTPL			
Estudiante: Erika Tatiana Vasquez Tapia			
NRO.	PALABRA	TOKEN	TIPO
- 1	start	palabra	Palabra Reservada
- 2	double	palabra	Palabra Reservada
- 3	suma	palabra	Identificador
- 4	=	=	Asignacion
- 5	5,5	numero	Valor Decimal
ERROR, VERIFIQUE EL CODIGO			
- 6	;	;	Caracter Especial
- 7	int	palabra	Palabra Reservada
- 8	a	palabra	Identificador
- 9	=	=	Asignacion
- 10	5	numero	Valor Entero
ERROR, VERIFIQUE EL CODIGO			
- 11	;	;	Caracter Especial
- 12	int	palabra	Palabra Reservada
- 13	5	numero	Valor Entero
ERROR, VERIFIQUE EL CODIGO			
- 14	;	;	Caracter Especial
- 15	if	palabra	Palabra Reservada
- 16	(	(	Caracter Especial
- 17	numero	palabra	Identificador
- 18	>=	>=	Operador Relacional

**Fig.6/** Resultado de la Ejecución del Analizador Léxico, Código Incorrecto.

## Repositorio del Proyecto

[https://github.com/etvasquez/Analizador\\_Lexico/tree/master/Analizador%20Lexico](https://github.com/etvasquez/Analizador_Lexico/tree/master/Analizador%20Lexico)