MANUAL TÉCNICO

INDICE

PRESENTACIÓN

El siguiente manual describe los pasos necesarios para que cualquier persona que tenga cierta base de programación en Python, manejo de HTML y JavaScript pueda hacer uso y entender el funcionamiento del programa, a su vez guiara a los usuarios que harán soporte al sistema, el cual les dará a conocer los requerimientos y la estructura para la construcción del sistema, además contiene descripción de las expresiones regulares del lenguaje desarrollado.

OBJETIVOS

General:

- Brindar información necesaria para el uso y manejo del programa, con el fin de que se pueda hacer soporte y modificaciones en caso de ser necesarias

Específicos:

1. Describir los métodos creados en el programa
2. Mostrar las expresiones regulares utilizadas para reconocer el lenguaje

PROCESOS

Procesos de entrada:

Ingresar al programa

Seleccionar el archivo del archivo .form

Procesos de salida:

Reporte de toques

Reporte de errores

Formulario

Manual de usuario

Manual técnico

DESCRIPCION DE LOS METODOS CREADOS

Clase Menú

- **Analizar()**
  Este método tomara el contenido del archivo .form y lo analizara para mostrar el formulario, si se encuentra un error léxico no se generara el formulario y se mostrara un mensaje de error.
- **Cargar_archivo()**
  Este método abrirá una ventana emergente para que se pueda elegir el archivo .form y mostrar su contenido en la interfaz, si no tiene la extensión correcta no se cargara nada y se mostrara un mensaje de error.
- **Reporte de tokens()**
  Analizara todo el contenido y mostrara un HTML con los tokens encontrados en el archivo cargado
- **Reporte de errores()**
  Analizara todo el contenido y mostrara un HTML con los errores encontrados en el archivo cargado.
- **Manual de usuario()**
  Mostrará el manual de usuario en formato .pdf
- **Manual técnico()**
  Mostrará el manual técnico en formato .pdf

Clase Tokens

Esta clase solamente contiene las variables lexema, línea, columna y tipo que servirán para crear un objeto error para posteriormente agregarlo a una lista.

Clase Reporte_errores

- **Análisis()**
  Analizara el contenido del archivo usando un autómata finito determinista y creara una lista con los errores encontrados.
- **Html_Errores()**
  Creará una HTML con los errores encontrados y se mostrara al usuario.

Clase Reporte_tokens()

- **Análisis()**
  Analizara el contenido del archivo usando un autómata finito determinista y creara una lista con los tokens encontrados.
- **Html_tokens()**
  Creará un HTML con los tokens encontrados y se mostrara al usuario.

Clase elementos:

Esta clase sirve para crear un objeto que tenga el valor y la variable que se usara para la creación del formulario.

Clase Error

Esta clase sirve para crear un objeto error que tenga la descripción, línea y columna para poder crear el HTML de reporte de errores.

Clase Formulario

- **Crear()**
  Este método guardara los valores y variables que servirán para la creación del formulario y los agregara en una lista.
- **Crear_componentes()**
  Este método creara los componentes tomándolo de la lista creada en el método “Crear” y los agregara a otra lista.
- **Crear_formulario()**
  Este método tomara los componentes de la lista creada en el método anterior y creara el formulario en HTML y lo mostrara al usuario.

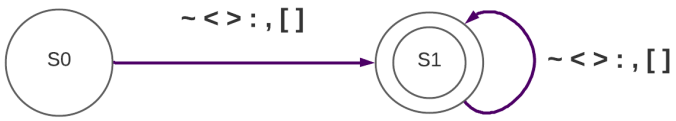
L	[a-zA-Z]
N	[0-9]
R	[Cualquier símbolo]

EXPRESIONES REGULARES

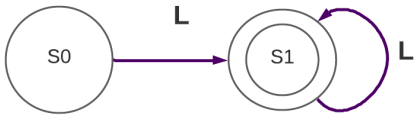
Token	**Expresión regular**
Símbolos	(\~|\<|\>|\:|\,|\[|\])+
Variable	L+
Palabra reservada	formulario
Cadena 1	“(L|N|R)*”
Cadena 2	‘(L|N|R)*’

AFD PARA CADA TOKEN

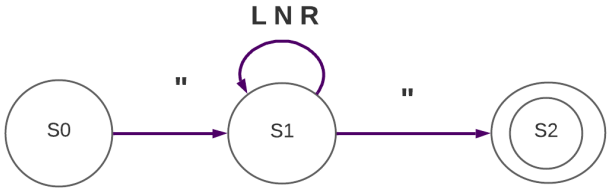
Token simbolo (AFD)



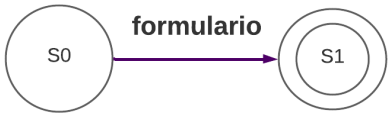
Token variable (AFD)



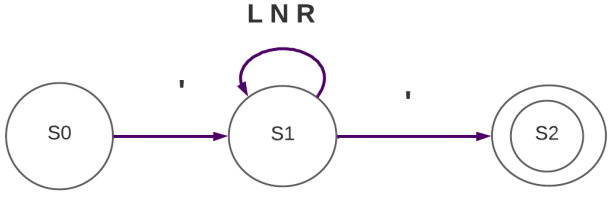
Token cadena 1 (AFD)



Token palabra reservada (AFD)



Token cadena 2 (AFD)



ARBOL PARA EXPRESION GENERAL

Expresión general
[(\~|\<|\>|\-|\:|\,)+|L+|formulario|(“(L|N|R)*”|‘(L|N|R)*’]#

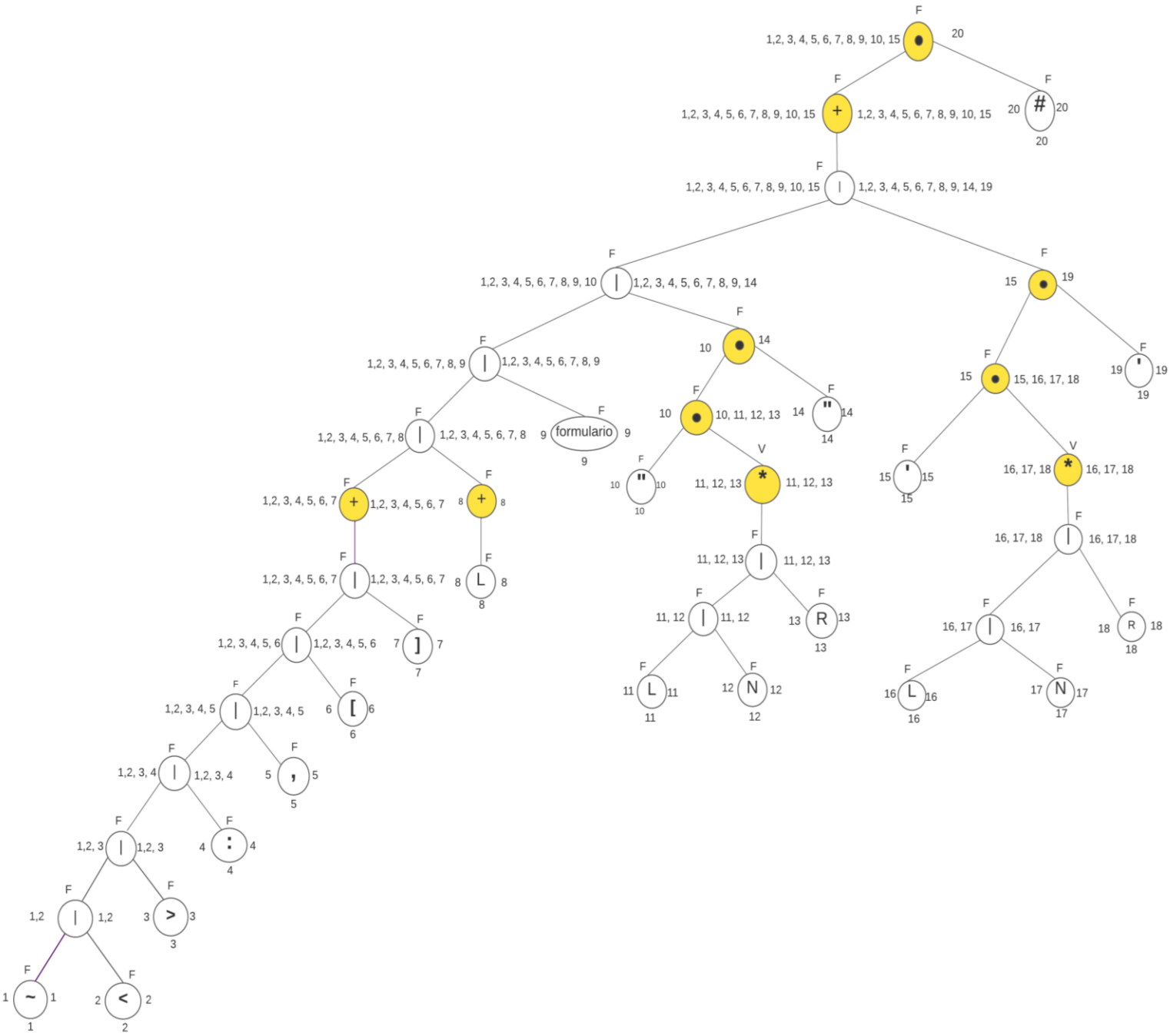


TABLA DE SIGUIENTES

Símbolo	No.	Siguiente
~	1	1 2 3 4 5 6 7 8 9 10 15 20
<	2	1 2 3 4 5 6 7 8 9 10 15 20
>	3	1 2 3 4 5 6 7 8 9 10 15 20
.	4	1 2 3 4 5 6 7 8 9 10 15 20
,	5	1 2 3 4 5 6 7 8 9 10 15 20
[	6	1 2 3 4 5 6 7 8 9 10 15 20
]	7	1 2 3 4 5 6 7 8 9 10 15 20
L	8	1 2 3 4 5 6 7 8 9 10 15 20
Formulario	9	1 2 3 4 5 6 7 8 9 10 15 20
“	10	11 12 13 14
L	11	11 12 13 14
N	12	11 12 13 14
R	13	11 12 13 14
“	14	1 2 3 4 5 6 7 8 9 10 15 20
‘	15	16 17 18 19
L	16	16 17 18 19
N	17	16 17 18 19
R	18	16 17 18 19
‘	19	1 2 3 4 5 6 7 8 9 10 15 20
#	20	

TABLA DE TRANSICIONES

S0 = {1 2 3 4 5 6 7 8 9 10 15}			
Símbolo	**Siguiente**		**Estado**
~ 1	1	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
2 <	2	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
3 >	3	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
4 :	4	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
5 ,	5	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
6 [	6	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
7]	7	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
8 L	8	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
9 Formulario	9	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
10 “	10	11 12 13 14	**S2**
15 ‘	15	16 17 18 19	**S3**

S1 = {1 2 3 4 5 6 7 8 9 10 15, 20} Estado de aceptación			
Símbolo	**Siguiente**		**Estado**
~ 1	1	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
2 <	2	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
3 >	3	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
4 :	4	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
5 ,	5	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
6 [	6	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
7]	7	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
8 L	8	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
9 Formulario	9	1 2 3 4 5 6 7 8 9 10 15 20	**S1**
10 “	10	11 12 13 14	**S2**
15 ‘	15	16 17 18 19	**S3**

S2 = {11 12 13 14}			
Símbolo	**Siguiente**		**Estado**
L 11	11	11 12 13 14	**S2**
N 12	12	11 12 13 14	**S2**
R 13	13	11 12 13 14	**S2**
“ 14	14	1 2 3 4 5 6 7 8 9 10 15 20	**S1**

S3 = {16 17 18 19}			
Símbolo	**Siguiente**		**Estado**
L 16	16	16 17 18 19	**S3**
N 17	17	16 17 18 19	**S3**
R 18	18	16 17 18 19	**S3**
‘ 19	19	1 2 3 4 5 6 7 8 9 10 15 20	**S1**

AUTOMATA FINITO DETERMINISTA