UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

LENGUAJES FORMALES DE PROGRAMACION

SECCION: B+

PROYECTO 1

Nombre: Jonatan Leonel Garcia Arana

Carné: 202000424

Fecha de entrega: 22/09/2022

OBJETIVOS

• Crear una solución de software usando un analizador léxico.

• El uso e implementación de un autómata y expresiones regulares

• Comprender el manejo de árboles y AFD

• Emplear el lenguaje de Python

ALCANCES

• El sistema se diseñó para sea capaz de ejecutar un archivo de entrada el cual este compuesto de tokens los que clasifican las palabras.

• Genera reportes en HTML con los datos procesados del software.

• Genera reportes en HTML con los errores procesados del software.

INFORMACION DEL SISTEMA

Requisitos del sistema:

• Windows 10,8,7 (x86 y x64)

• Procesador a 1.6 GHz o superior

• 1 GB (32 bits) o 2 GB (64 bits) de RAM (agregue 512 MB al host si se ejecuta en una máquina virtual)

• 3 GB de espacio disponible en el disco duro

• Disco duro de 5400 RPM

• Tarjeta de vídeo compatible con DirectX 9 con resolución de pantalla de 1024 x 768 o más.

Descargar Visual studio Code link de descarga:

https://code.visualstudio.com/download

ver manual de instalación en la página Oficial Descargar Python
https://www.python.org/downloads/

ver manual de instalación en la página Oficial

CLASES y METODOS UTILIZADOS

LEXICO ():

Esta clase contiene un método que es para analizar la cadena y por medio de la expresión regular crear los estados los cuales nos servirán para complementar para manipular la información.

ARIMETICAS ():

En esta clase mandamos a hacer las operaciones que extraemos del archivo, utilizando recursividad

OPERADOR ():

Esta clase es la contiene el de operación que se encuentre en el archivo

Abrir_archivo ():

Este método sirve para cargar el archivo de entrada y luego insertarla en el cuadro de texto

Guardar ():

Este método guarda el archivo con el mismo nombre

Leer_archivo ():

Este método es que manda a analizar el archivo

Guardar_como ():

Este método nos permite guardar el archivo con diferente nombre

Html_errores1 ():

En este método generamos el html de errores

Inicio ():

Este método es el que genera la ventana utilizando tkinter

FLUJO DEL PROGRAMA

DEFINIR LOS ELEMENTOS

S = ["<", ">", "=", “.”, "/", "[“, "]"]

R= [a-z, ñ]

A = [A-Z, Ñ,]

D = Dígitos [0-9]

Espacio = [“”]

NOMBRE	PATRON	EXPRESION REGULAR	EJEMPLO
RESERVADA	Inicia con una letra minúscula la cual puede seguir con 0 o cualquier combinación de letras	R+	-titulo -descripción -contenido
SIMBOLOS	Inicia con un símbolo y solo puede venir una vez	S	-< ->> -= -/
LITERALES	Inicia con una letra Mayúscula a la que le siguen letras minúsculas 1 o mas letras	AR+	Tipo Suma Resta
DIGITOS	Inicia con un numero o los que sean necesarios, luego puede venir o no un punto y más números	D+|D+(.)D+	- 1 - 2.4 - 22.44 - 223.523

EXPRESION REGULAR

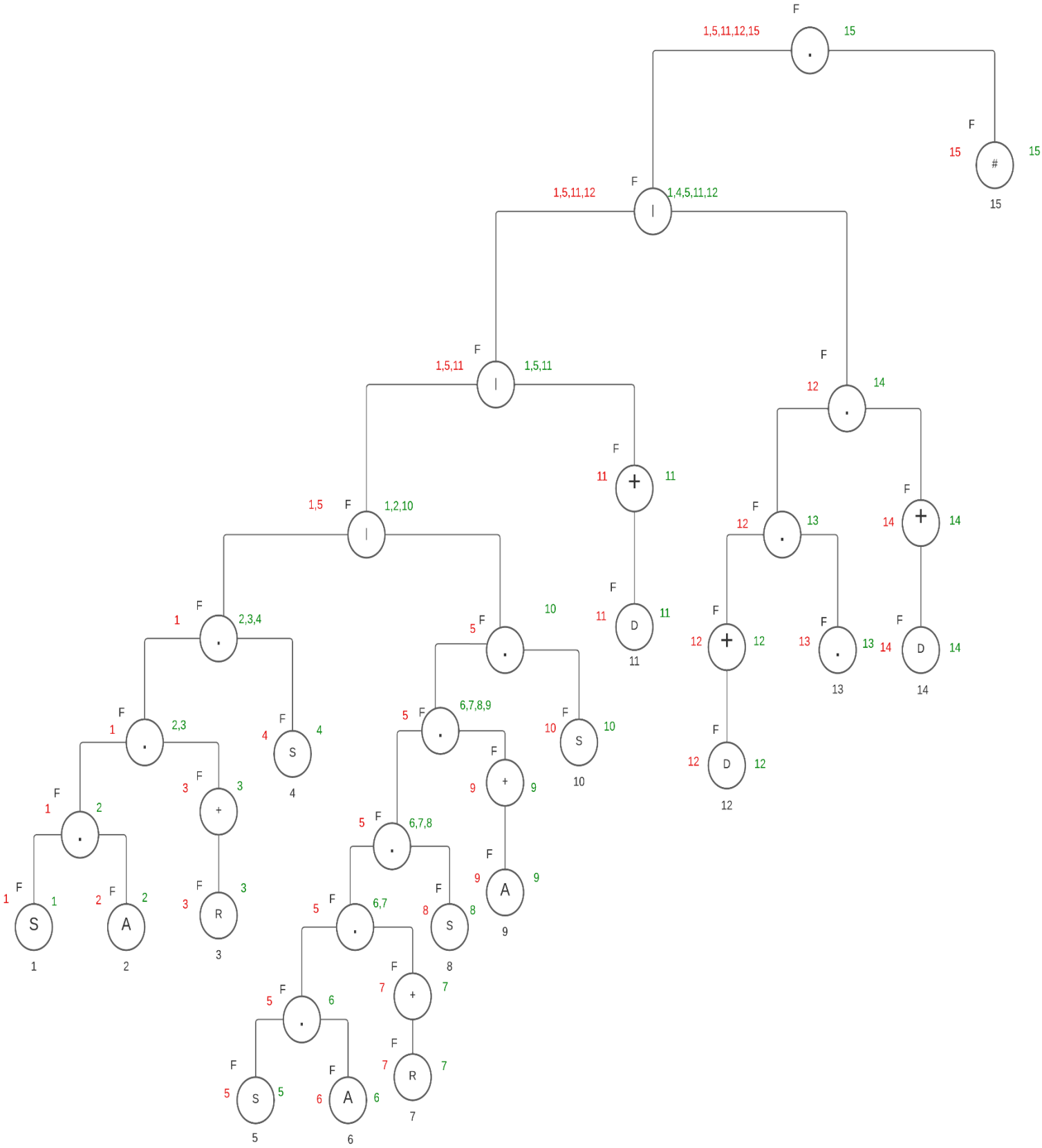
SAR+S | SAR+SA+S | D+|D+(.)D+

1. Concatenar símbolo de aceptación al final de la expresión (SAR+S | SAR+SA+S | D+|D+(.) D+)#

2. Construir el árbol binario de sintaxis

3. Identificar cada hoja con terminales.

4. Calcular por cada nodo del árbol: Anulable, First, Last.



5. Calcular Siguientes

VALOR	HOJAS	SIGUIENTES
S	1	1,2,3,4
A	2	2,3,4
R	3	3,4
S	4	15
S	5	5,6,7,8,9,10
A	6	6,7,8,9,10
R	7	7,8,9,10
S	8	8,9,10
A	9	9,10
S	10	15
D	11	15
D	12	13,14
.	13	14
D	14	15
#	15	--

6. Construir Tabla de Transiciones

	Estado	Valores	Siguientes
INICIO	S1	S,D,D 1,11,12	1: {2} = S2 D: {11} = S9 D:{12} = S11
	S2	A 2	A:{2} = S3
	S3	R 3	R:{3} = S4
	S4	S 4	S:{4} =S5 S:{4}=S6
#	S5	# 15	--
	S6	A 9	S{9} = S7
	S7	S 10	S{10}=S5
#	S8	D 11	--
	S9	D 12	D{12} = S10
	S10	S 13	S{13}= S11
#	S11	D 14	--

7. Tabla de Estados

ESTADOS		SIGMA				
		S	A	R	D	.
Inicio	S1	S2			S8,S10	
	S2		S3			
	S3			S4		
	S4	S5,S6				
#	S5					
	S6		S7			
	S7	S5				
#	S8				S8	
	S9					S10
	S10				S11	
#	S11					

AUTOMATA FINITO DETERMINISTAS