Universidad de San Carlos de Guatemala

Facultad de Ingeniería

PROYECTO 2

MANUAL TECNICO

Nathan Antonio Valdez[2]

202001568

Sección B+

LABORATORIO LENGUAJES FORMALES Y DE PROGRAMACION

Objetivos y alcances del sistema

- Que el usuario pueda ingresar a un chat inteligente de respuestas automáticas y por medio de comandos pueda preguntarle información acerca de resultado de partidos, descripción completa de jornadas, goles de un equipo, tablas de clasificación por temporadas, partidos por rango de jornadas, etc.

- El sistema esta diseñado con una interfaz amigable para el usuario en donde solo se encuentra una ventana con los mensajes enviados al bot y los que el bot responde, una caja de texto donde el usuario puede ingresar los comandos especificados mas abajo, un botón de enviar y un botón de mas opciones posicionado en la esquina superior derecha.

- Crear un analizador léxico utilizando los conceptos de alfabeto, tokens y sus propiedades

- Desarrollar una interfaz gráfica utilizando el lenguaje de programación Python

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

Librerias Utilizadas:

Tkinter:

https://docs.python.org/3/library/tkinter.html

jinja2:
https://svn.python.org/projects/external/Jinja2.1.1/docs/_build/html/index.html

csv:

https://docs.python.org/3/library/csv.html

ANALIZADOR LEXICO

Elementos:

L = [A-Z]

d = [0-9]

p = [fjnif]

Nombre	Patron	Expresión Regular	Ejemplo
RESERVADA	Inicia con una letra mayúscula ala que le siguen cero o muchas veces cualquier combinación de letras mayúsculas	L(L)*	✓ RESULTADO ✓ JORNADA ✓ GOLES
STRINGS	Inicia con comillas dobles a la que le siguen una o muchas veces cualquier carácter hasta encontrar otras comillas dobles	"(^")*"	✓ “Real Madrid” ✓ “Barcelona”
TEMPORADA	Inicia con el símbolo menor que a la que le siguen 4 digitos seguido de un guion seguido de 4 digitos y finalizando con el símbolo mayor que	<d+-d+>	✓ <2019-2020> ✓ <2001-2002> ✓ <1974-1975>
BANDERA	Inicia con el símbolo menor al que le sigue cualquier carácter del conjunto p una o dos veces	-(p){1,2}	✓ -f ✓ -ji ✓ -jf ✓ -n
NUMERO	Inicia con un digito al que le puede seguir otro digito o no	dd?	✓ 1 ✓ 23 ✓ 2
NAME	Inicia con una letra mayúscula o minúscula a la que le sigue cero o muchas veces cualquier combinación de letras, y le sigue cero o una vez un sub-guion o dos puntos.	\w+	✓ Reporte1 ✓ reporteEspanol

Universidad de San Carlos de Guatemala

Facultad de Ingeniería

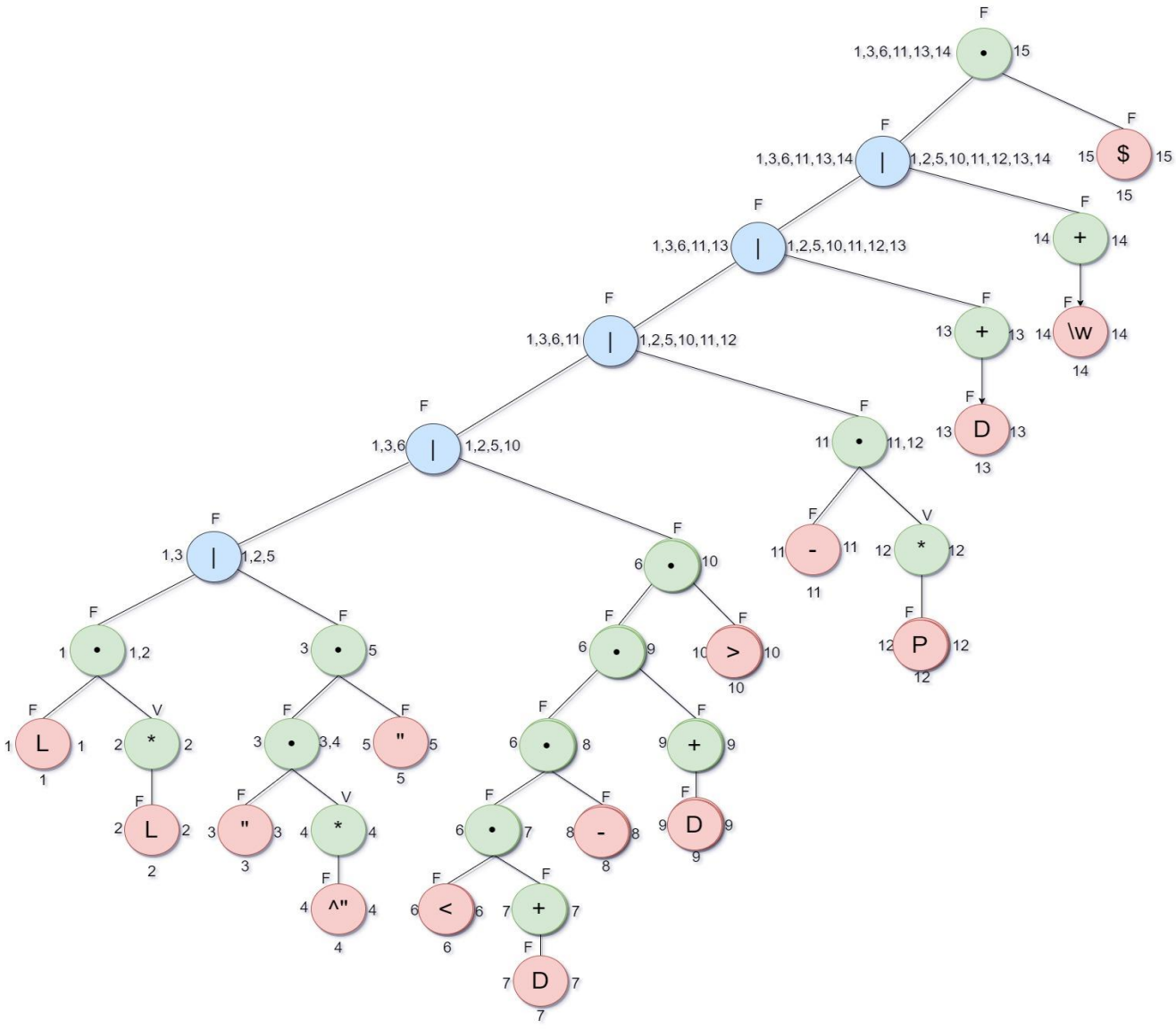
ANALIZIS LÉXICO

L(L)*|"(^")*")|<d+-d+>|-(p)*|dd?|\w+

1. **Concatenar símbolo de aceptación al final de la ER**

(L(L)*|"(^")*")|<d+-d+>|-(p)*|dd?|\w+)$

2. **Construir el árbol binario de sintaxis.**
3. **Identificar cada hoja con terminales.**
4. **Calcular por cada nodo del árbol: Anulable, First, Last.**



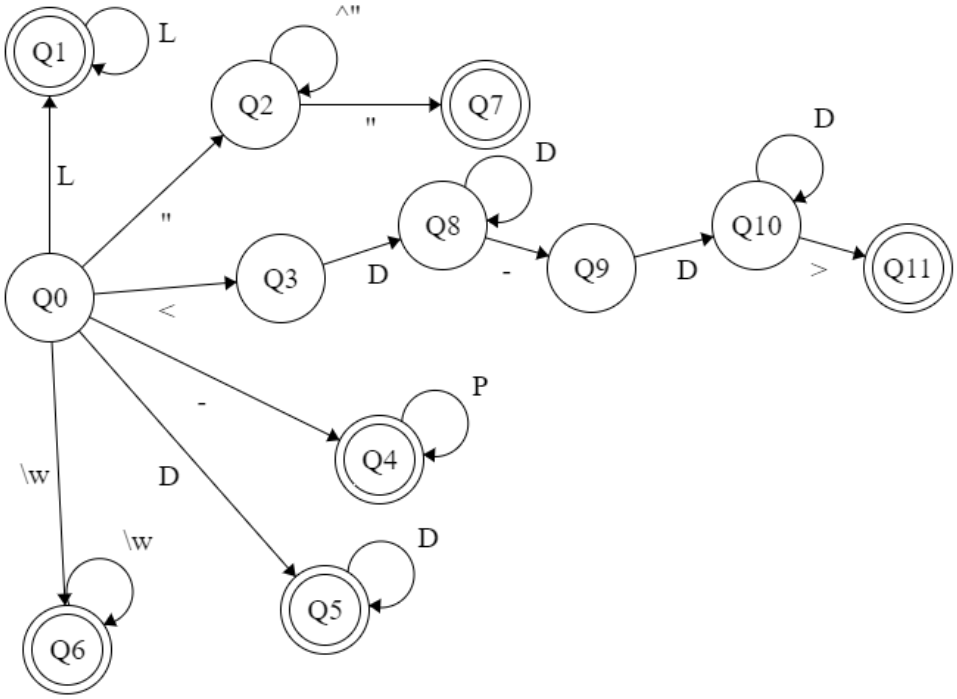
5. **Calcular Siguientes**

HIJOS	HOJA	SIGUIENTE
L	1	2,15
L	2	2,15
“	3	4,5
^”	4	4,5
“	5	15
<	6	7
D	7	7,8
-	8	9
D	9	9,10
>	10	15
-	11	12,15
P	12	12,15
D	13	13,15
\w	14	14,15
$	15	----------

6. **Tabla de transiciones**

	ESTADO	VALORES	SIGUIENTE
O	Q0	1,3,6,11,13,14 L, ", <, -, D,\W	L = {2,15} = Q1
			" = {4,5} = Q2
			< = {7} = Q3
			MENOS = {12,15} = Q4
			D = {13,15} = Q5
			\W = {14,15} = Q6
$	Q1	2,15	L = {2,15} = Q1
		L, $	
	Q2	4, 5	^" = {4,5} = Q2
		^", "	" = {15} Q7
	Q3	7	D = {7,8} = Q8
		D	
$	Q4	12, 15	P = {12,15} = Q4
		P, $	
$	Q5	13, 15	D = {13,15} = Q5
		D , $	
$	Q6	14, 15	\W = {14,15} = Q6
		\W, $	
$	Q7	15	
		$	
	Q8	7,8	D = {7,8} = Q8
		D, -	MENOS = {9} = Q9
	Q9	9	D = {9,10} = Q10
		D	
	Q10	9,10	D = {9,10} = Q10
		D, >	> = {15} = Q11
$	Q11	15	$$$$$$$$$$$$
		$	

7. Autómata Finito Determinista



ANALIZIS SINTACTICO

<Inicio> : : = <RESULTADO>
| <JORNADA>
| <GOLES>
| <TABLA>
| <PARTIDOS>
| <TOP>
| <ADIOS>

<RESULTADO> : : = res_resultado string res_vs string res_temporada tk_tmp

<JORNADA> : : = res_jornada tk_num res_temporada tk_temp <JORNADA'>
<JORNADA'> : : = tk_flag tk_name
| épsilon

<GOLES> : : = res_goles <COND1> string res_temporada tk_tmp
<COND> : : = res_local
| res_visitante
| res_total

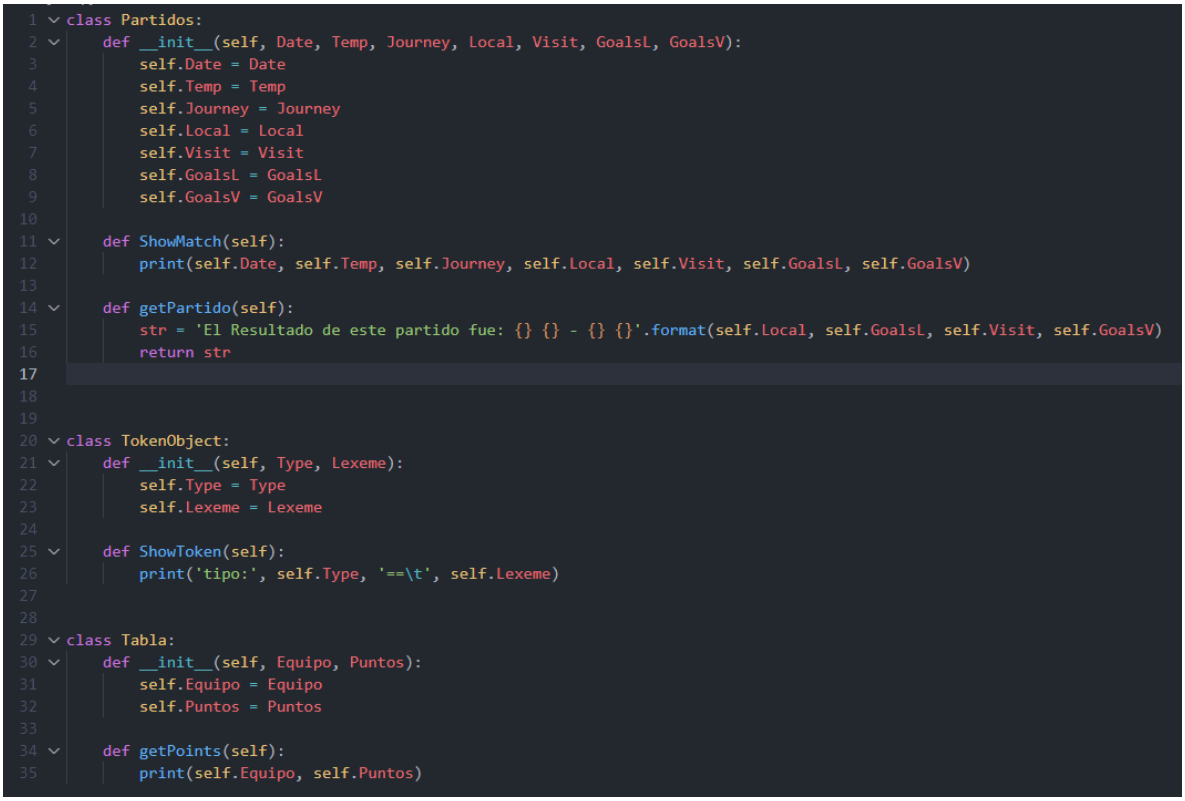
<TABLA> : : = res_tabla res_temporada tk_temp <TABLA'>
<TABLA'> : : = tk_flag tk_num
| épsilon

<PARTIDOS> : : = res_partidos string res_temporada tk_temp <PARTIDOS'>
<PARTIDOS'> : : = tk_FLAGF tk_ID tk_FLAGJI tk_num tk_FLAGJF tk_num
| tk_FLAGF tk_ID tk_FLAGJI tk_num
| tk_FLAGJI tk_num tk_FLAGJF tk_num
| tk_FLAGF tk_ID tk_FLAGJF tk_num
| épsilon

<TOP> : : = res_top <COND2> res_temporada tk_temp <TOP'>
<COND2> : : = res_superior
| res_inferior
<TOP'> = tk_FLAGN tknum
| épsilon

<ADIOS> : : = res_adios

OBJETOS UTILIZADOS (POO)



```python
class Partidos:
    def __init__(self, Date, Temp, Journey, Local, Visit, GoalsL, GoalsV):
        self.Date = Date
        self.Temp = Temp
        self.Journey = Journey
        self.Local = Local
        self.Visit = Visit
        self.GoalsL = GoalsL
        self.GoalsV = GoalsV

    def ShowMatch(self):
        print(self.Date, self.Temp, self.Journey, self.Local, self.Visit, self.GoalsL, self.GoalsV)

    def getPartido(self):
        str = 'El Resultado de este partido fue: {} {} - {} {}'.format(self.Local, self.GoalsL, self.Visit, self.GoalsV)
        return str


class TokenObject:
    def __init__(self, Type, Lexeme):
        self.Type = Type
        self.Lexeme = Lexeme

    def ShowToken(self):
        print('tipo:', self.Type, '==\t', self.Lexeme)


class Tabla:
    def __init__(self, Equipo, Puntos):
        self.Equipo = Equipo
        self.Puntos = Puntos

    def getPoints(self):
        print(self.Equipo, self.Puntos)
```

1. **PARTIDOS:**
   Este objeto se utiliza al inicio del programa para leer el archivo de entrada CSV y almacenar cada partido que este contiene, en este se guarda la fecha del partido, temporada en que se jugó, numero de jornada, equipo local, equipo visitante, goles del equipo local y goles del equipo visitante

2. **TokenObject:**
   En este objeto se guardan todos los tokens que se han guardado durante la ejecución del programa, así como los errores encontrados

3. **Tabla:**
   Este objeto es utilizado a la hora de generar tablas en los reportes, solamente se almacena el nombre del equipo y su puntuación durante la temporada