Universidad Rafael Landívar
Facultad de ingeniería
Compiladores
Ing. Juan Carlos Soto Santiago

Yet **A**nother **YACC**

ENTREGA 1

Kevin Humberto Romero Villalta 1047519
Jocelyn Abigail de León Ardón 1305619
José Vinicio De León Jiménez 1072619

Guatemala, 1 de noviembre de 2021

Analizador Léxico

Token	Patron	Descripción
T_NonT	**([A-Z]|[a-z]|_)+(\w)*** http://www.regexr.com/68m12	Para las variables en las gramáticas yayacc.
T_Terminal	**'(\w|!|"|#|%|&|\(|\)|*|\+|,|-|\.|\/|:|;|<|=|>|\?|\[|\]|\^|_|{|||}|\\n|\\t|\\\|\')+'+** http://www.regexr.com/68m0p	Para las cadenas terminales en las gramáticas yayacc.
T_Or	|	El separador de reglas para una misma variable.
T_Separator	:	Para la definición de una variable y su regla asociada.
T_End	;	Para describir el fin de una regla simple o compuesta.
T_Enter	**\n**	Para describir una nueva línea en la gramática.

Analizador Sintáctico

- Gramática Base
  1. E' -> S'
  2. S' -> S ; \n S'
  3. S' -> S ;
  4. S -> non_t : rule
  5. S -> non_t : rule | D
  6. D -> rule | D
  7. D -> rule
  8. rule -> rule rule
  9. rule -> terminal
  10. rule -> non_t

- Tabla LALR

	ACTIONS							GOTO				
ESTADOS	;	\n	non_t	:	|	terminal	$	E'	S'	S	D	rule
0			S3						1	2		
1							ACCEPTED					
2	S4											
3				S5								
4		S6					R2					
5			S9			S8						7
6			S3						10	2		
7	R3		S9		S11	S8						12
8	R8		R8		R8	R8						
9	R9		R9		R9	R9						
10							R1					
11			S9			S8					13	14
12	R7		S9		R7	S8						12
13	R4											
14	R6		S9		S15	S8						12
15			S9			S8					16	14
16	R5											

Ver GrafoLALR.pdf adjunto.

Gramáticas en YaYacc

1. Misma cantidad de 0's que de 1's

```
S' : S ;
S : '0' S '1 ' | '01' ;
```

Luego de analizar léxicamente el archivo .yayacc obtenemos el siguiente flujo de tokens.

```
non_t : non_t ; \n
non_t : terminal non_t terminal | terminal ;
```

Luego el analizador sintáctico toma dichos tokens y con base a la gramática principal valida la gramática yayacc ingresada.

Ver pág. Gramática 1 del Excel adjunto.

2. Formación de oraciones simples

```
S' : S ;
S : A B ;
A : 'the' ;
B : C 'cat' ;
C : 'happy' ;
C : 'sad' ;
```

Luego de analizar léxicamente el archivo .yayacc obtenemos el siguiente flujo de tokens.

```
non_t : non_t ; \n
non_t : non_t non_t ; \n
non_t : terminal ; \n
non_t : non_t terminal ; \n
non_t : terminal ; \n
non_t : terminal ;
```

Luego el analizador sintáctico toma dichos tokens y con base a la gramática principal valida la gramática yayacc ingresada.

Ver pág. Gramática 2 del Excel adjunto.

3. Expresiones de a's que inician con + o *

```
S' : S;
S : '+' SS | '*' SS | 'a' ;
```

Luego de analizar léxicamente el archivo .yayacc obtenemos el siguiente flujo de tokens.

non_t : non_t ; \n
non_t : terminal non_t non_t | terminal non_t non_t | terminal ;

Luego el analizador sintáctico toma dichos tokens y con base a la gramática principal valida la gramática yayacc ingresada.

Ver pág. Gramática 3 del Excel adjunto.

4. Expresiones aritméticas con sumas y multiplicaciones

```
S' : S ;
S : S '+' T | T ;
T : T '*' F | F ;
F : '(' S ')' | 'num' ;
```

Luego de analizar léxicamente el archivo .yayacc obtenemos el siguiente flujo de tokens.

non_t : non_t ; \n
non_t : non_t terminal non_t | non_t ; \n
non_t : non_t terminal non_t | non_t ; \n
non_t : terminal non_t terminal | terminal ;

Luego el analizador sintáctico toma dichos tokens y con base a la gramática principal valida la gramática yayacc ingresada.

Ver pág. Gramática 4 del Excel adjunto.

5. Paréntesis con a's separadas por comas

```
S’ : S ;
S : '(' L ')' | 'a' ;
L : L ',' S | S ;
```

Luego de analizar léxicamente el archivo .yayacc obtenemos el siguiente flujo de tokens.

non_t : non_t ; \n
non_t : terminal non_t terminal | terminal ; \n
non_t : non_t terminal non_t | non_t ;

Luego el analizador sintáctico toma dichos tokens y con base a la gramática principal valida la gramática yayacc ingresada.

Ver pág. Gramática 5 del Excel adjunto.

6. Expresiones Booleanas

```
S’ : E ;
E : E 'or' T | T ;
T : T 'and' F | F ;
```

```
F : 'not' F | '(' E ')' ;
F : 'true' | 'false' ;
```

Luego de analizar léxicamente el archivo .yayacc obtenemos el siguiente flujo de tokens.

non_t : non_t ; \n
non_t : non_t terminal non_t | terminal ; \n
non_t : non_t terminal non_t | non_t ; \n
non_t : terminal non_t | terminal non_t terminal ; \n
non_t : terminal | terminal ;

Luego el analizador sintáctico toma dichos tokens y con base a la gramática principal valida la gramática yayacc ingresada.

Ver pág. Gramática 6 del Excel adjunto.

7. Firma de un método

```
S’ : S ;
S :  'MethodID' '(' P ')' ;
P : D ', ' P | D;
D : T 'id';
T : 'string' | 'int';
```

Luego de analizar léxicamente el archivo .yayacc obtenemos el siguiente flujo de tokens.

non_t : non_t ; \n
non_t : terminal terminal non_t terminal ; \n
non_t : non_t terminal non_t | non_t ; \n
non_t : non_t terminal ; \n
non_t : terminal | terminal ;

Luego el analizador sintáctico toma dichos tokens y con base a la gramática principal valida la gramática yayacc ingresada.

Ver pág. Gramática 7 del Excel adjunto.

8. Asignación de variables

```
S’ : S ;
S : 'id' | V '=' E;
V : 'id ';
E : V | 'n';
```

Luego de analizar léxicamente el archivo .yayacc obtenemos el siguiente flujo de tokens.

non_t : non_t ; \n

non_t : terminal | non_t terminal non_t ; \n
non_t : terminal ; \n
non_t : non_t | terminal ;

Luego el analizador sintáctico toma dichos tokens y con base a la gramática principal valida la gramática yayacc ingresada.

Ver pág. Gramática 8 del Excel adjunto.

9. Palabras de tres caracteres

```
S’ : S ;
S : 'a' A 'd' | 'b' A 'd' | 'a' A 'e' | 'b' A 'e';
A : 'c';
```

Luego de analizar léxicamente el archivo .yayacc obtenemos el siguiente flujo de tokens.

non_t : non_t ; \n
non_t : terminal non_t terminal | terminal non_t terminal | terminal non_t terminal | terminal non_t terminal ; \n
non_t : terminal ;

Luego el analizador sintáctico toma dichos tokens y con base a la gramática principal valida la gramática yayacc ingresada.

Ver pág. Gramática 9 del Excel adjunto.

10. Paréntesis bien balanceados simples

```
S' : S ;
S : '(' S ') ' | '()' ;
```

Luego de analizar léxicamente el archivo .yayacc obtenemos el siguiente flujo de tokens.

non_t : non_t ; \n
non_t : terminal non_t terminal | terminal ;

Luego el analizador sintáctico toma dichos tokens y con base a la gramática principal valida la gramática yayacc ingresada.

Ver pág. Gramática 10 del Excel adjunto.