

5.pdf



Anónimo



Procesadores de Lenguajes



3º Grado en Ingeniería Informática



**Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid**



Descarga la APP de Wuolah.
Ya disponible para el móvil y la tablet.





**KEEP
CALM
AND
ESTUDIA
UN POQUITO**

27-JUNIO-2018

Una gramática regular para el fragmento de lenguaje dado es:

$A \rightarrow l B \mid = \mid < C \mid > F \mid \text{del } A$
 $B \rightarrow d B \mid l B \mid \lambda$
 $C \rightarrow \lambda \mid > \mid = \mid < D$
 $D \rightarrow c D \mid > E$
 $E \rightarrow > A \mid c D$
 $F \rightarrow = \mid \lambda$

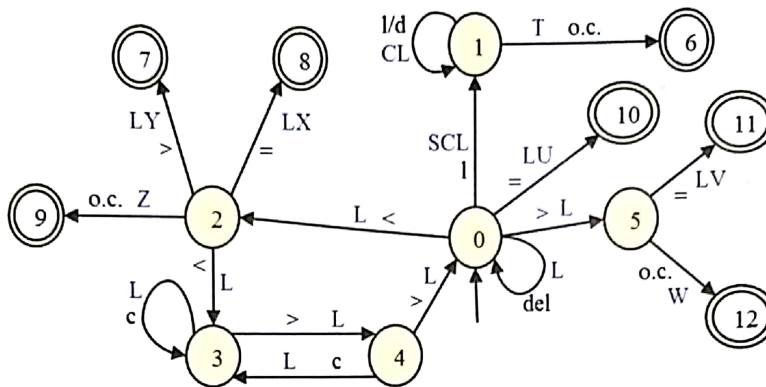
Siendo:

c: cualquier carácter excepto '>'
 del: blanco, tabulador o salto de línea
 l: cualquier letra
 d: cualquier dígito

Los tokens para este lenguaje serían:

- <ID, posTS>: representa un identificador, siendo el atributo su entrada en la tabla de símbolos
- <OPR, n>: representa un operador relacional, siendo el atributo n un valor correspondiente al operador leído (1: =, 2: >=, 3: >, 4: <=, 5: <,>, 6: <)

El autómata finito determinista correspondiente a este lenguaje sería:



Por último, el AFD se completa con las acciones semánticas, donde:

- GenToken: genera un token con los parámetros recibidos (nótese que los comentarios no generan token)
- lee: lee el siguiente carácter del fichero de entrada
- Concat: concatena los caracteres del identificador
- BuscaTS: busca en la tabla de símbolos la palabra recibida y devuelve su posición
- InsertaTS: introduce en la tabla de símbolos la palabra recibida y devuelve su posición
- Error: genera un mensaje de error. Cualquier transición no indicada en el autómata corresponde a un caso de error léxico, que daría lugar a un mensaje. Así, en el estado 0, si se encuentra otro carácter, debería dar un mensaje del tipo "Carácter no válido al inicio de un elemento del lenguaje".

L:	Lee
C:	Concat(Pal); Cont:= Cont + 1
S:	Pal:=∅; Cont:=0
T:	If (Cont ≤ 256) Then { p:= BuscaTS (Pal) if (p=NULL) Then p:= InsertaTS (pal) GenToken (ID, p) } else Error ("El identificador ", Pal, " tiene más de 256 caracteres")
U:	GenToken (OPR, 1)
V:	GenToken (OPR, 2)
W:	GenToken (OPR, 3)
X:	GenToken (OPR, 4)
Y:	GenToken (OPR, 5)
Z:	GenToken (OPR, 6)

27 de junio de 2018

WUOLAH

F → function id (R) { S }

Descarga la app de Wuolah desde tu store favorita

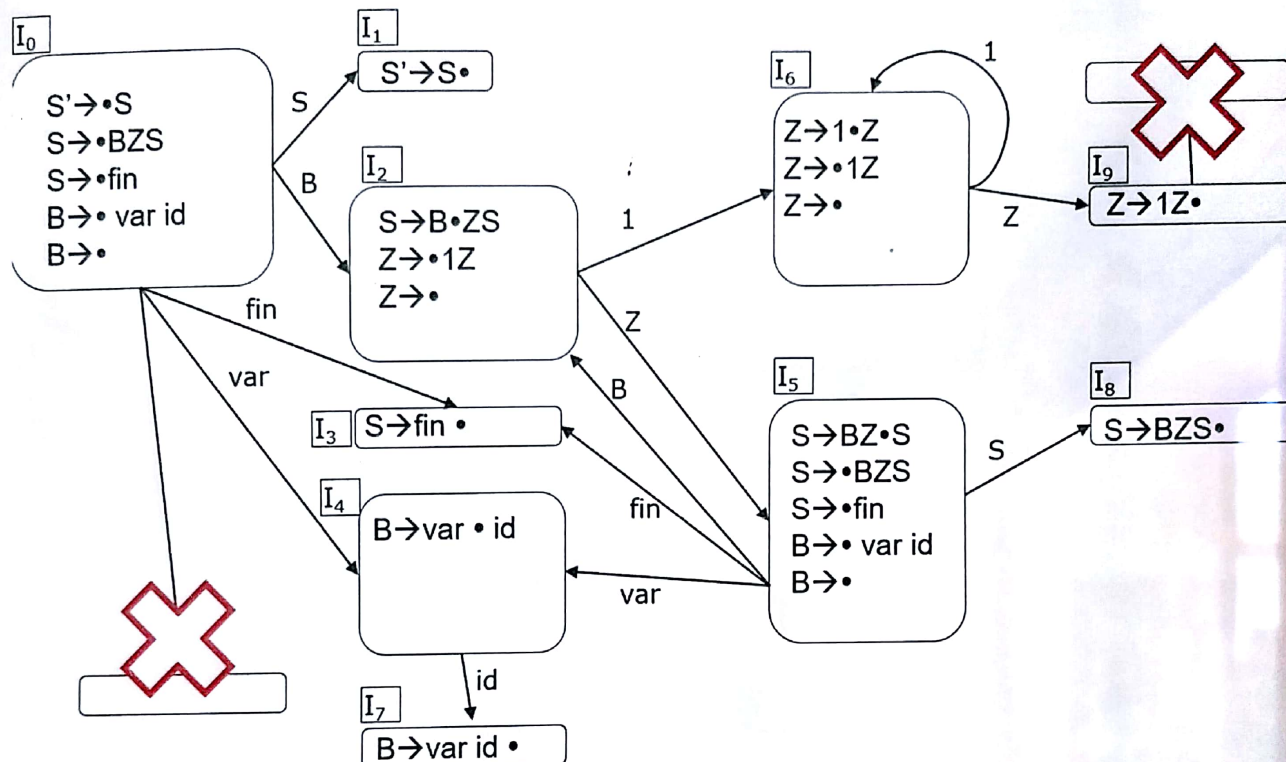
PROCESADORES DE LENGUAJES

Plantilla de Respuesta - Análisis Sintáctico. 27 de junio de 2018

En relación con el método de Análisis Sintáctico Ascendente (LR), dada la gramática:

$$\begin{aligned} S &\rightarrow B Z S \mid \text{fin} \\ B &\rightarrow \text{var id} \mid \lambda \\ Z &\rightarrow 1 Z \mid \lambda \end{aligned}$$

Se pide construir el **Autómata Reconocedor de Prefijos Viables** para esta gramática, usando la siguiente plantilla, que puede contener errores u omisiones:



Se pide realizar el análisis de todos los posibles **conflictos** en el Autómata resultante del apartado a.

- Follow(B) = {1, var, fin} Follow(Z) = {1, var, fin}
- I_0 Red-Desp Reducción por Follow(B); Desplazamiento por {var, fin}
por Follow(B) \cap {var, fin} $\neq \emptyset$ Hay Conflicto Reducción-Desplazamiento
- I_2 Red-Desp Reducción por Follow(Z); Desplazamiento por {1}
por Follow(Z) \cap {1} $\neq \emptyset$ Hay Conflicto Reducción-Desplazamiento
- I_5 Red-Desp Reducción por Follow(B); Desplazamiento por {var, fin}
por Follow(B) \cap {var, fin} $\neq \emptyset$ Hay Conflicto Reducción-Desplazamiento
- I_6 Red-Desp Reducción por Follow(Z); Desplazamiento por {1}
por Follow(Z) \cap {1} $\neq \emptyset$ Hay Conflicto Reducción-Desplazamiento

En relación con el método de **Análisis Sintáctico Descendente LL(1)**, dada la gramática:

$$\begin{aligned} A &\rightarrow 1 B \mid 2 C \mid D C \\ B &\rightarrow 1 C \end{aligned}$$

En relación con el método de Análisis Sintáctico Descendente LL(1), dada la gramática:

$$\begin{aligned} A &\rightarrow 1 B \mid 2 C \mid D C \\ B &\rightarrow 1 C \\ C &\rightarrow 3 D \mid 4 C \\ D &\rightarrow 5 A \mid 6 \mid \lambda \end{aligned}$$

Se pide corregir los errores u omisiones de la siguiente tabla LL(1):

	1	2	3	4	5	6	\$	λ
A	A \rightarrow 1 B	A \rightarrow 2 C	A \rightarrow D C	A \rightarrow D C	A \rightarrow D C	A \rightarrow D C	A \rightarrow D C	
B	B \rightarrow 1 C							
C			C \rightarrow 3 D	C \rightarrow 4 C				
D			D \rightarrow λ	D \rightarrow λ	D \rightarrow 5 A	D \rightarrow 6	D \rightarrow λ	D \rightarrow λ

Tabla Correcta:

	1	2	3	4	5	6	\$
A	A \rightarrow 1 B	A \rightarrow 2 C	A \rightarrow D C	A \rightarrow D C	A \rightarrow D C	A \rightarrow D C	
B	B \rightarrow 1 C						
C			C \rightarrow 3 D	C \rightarrow 4 C			
D			D \rightarrow λ	D \rightarrow λ	D \rightarrow 5 A	D \rightarrow 6	D \rightarrow λ

Se pide escribir los procedimientos correspondientes a los símbolos B y D pertenecientes a un Analizador Sintáctico Descendente Predictivo Recursivo, dada la gramática del apartado c.

Procedure B ()

```

Begin
  Equipara_Token (1)
  C
End B

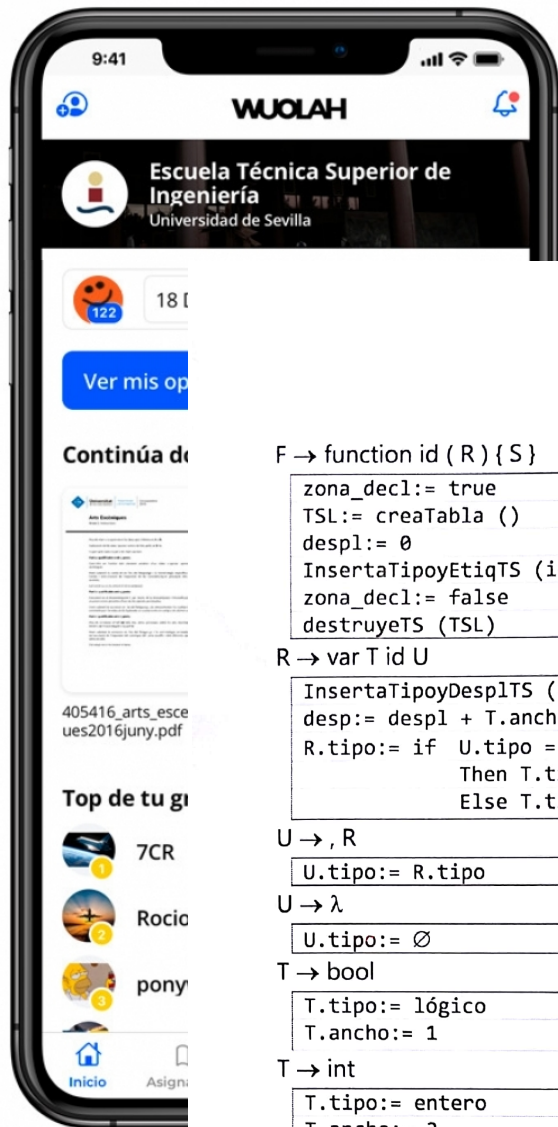
```

Procedure D ()

```

Begin
  If Sgte_token="5"
  Then Begin Equipara_Token (5)
        A
      End
  Else If Sgte_token="6"
  Then Equipara_Token (6)
  Else If Sgte_token  $\notin$  {3,4,$} //Follow(D)
  Then error ("Símbolo ", Sgte_Token, "inesperado")
End D

```



Descarga la APP de Wuolah.
Ya disponible para el móvil y la tablet.



27 de junio de 2018

F → function id (R) { S }

```
zona_decl:= true
TSL:= creaTabla ()
despl:= 0
InsertaTipoyEtiqTS (id.entrada, R.tipo→vacío, nuevaetiqa())
zona_decl:= false
destruyeTS (TSL)
```

R → var T id U

```
InsertaTipoyDesplTS (id.entrada, T.tipo, displ)
despl:= displ + T.ancho
R.tipo:= if U.tipo = Ø
Then T.tipo
Else T.tipo x U.tipo
```

U → , R

```
U.tipo:= R.tipo
```

U → λ

```
U.tipo:= Ø
```

T → bool

```
T.tipo:= lógico
T.ancho:= 1
```

T → chars

```
T.tipo:= cadena
T.ancho:= 10
```

T → int

```
T.tipo:= entero
T.ancho:= 2
```

T → real

```
T.tipo:= real
T.ancho:= 4
```

E → id

```
E.tipo:= BuscaTipoTS (id.entrada)
```

S → S₁; S₂

```
S.tipo:= if S1.tipo = tipo_ok
Then S2.tipo
Else tipo_error
```

S → call id (P)

```
S.tipo:= if BuscaTipoTS (id.entrada) = P.tipo→vacío
Then tipo_ok
Else tipo_error
```

S → id A E

```
id.tipo:= BuscaTipoTS (id.entrada)
S.tipo:= If A.op = 1 //operador +=
Then if id.tipo = E.tipo = lógico
Then tipo_ok
Else tipo_error
Else if A.op = 2 //operador %=
Then if id.tipo = E.tipo = cadena
Then tipo_ok
Else if id.tipo = E.tipo ∈ {entero, real}
Then tipo_ok
Else tipo_error
```

P → E

```
P.tipo:= E.tipo
```

P → E, P₁

```
P.tipo:= E.tipo x P1.tipo
```

A → +=

```
A.op:= 1
```

A → % =

```
A.op:= 2
```