

Análisis Sintáctico

Grupo 55

Aarón Cabero Blanco
Daniel Tomás Sanchez
Alejandro Cuadrón Lafuente

3 de diciembre de 2020

Índice

1. Diseño del Analizador Sintáctico	2
1.1. Gramática	2
1.2. Tabla LR	2
2. Anexo	6
2.1. Casos de prueba correctos	6
2.1.1. Caso de prueba 1	6
2.1.2. Caso de prueba 2	6
2.1.3. Caso de prueba 3	7
2.2. Casos de prueba incorrectos	8
2.2.1. Caso de prueba 1	8
2.2.2. Caso de prueba 2	8
2.2.3. Caso de prueba 3	9

1. Diseño del Analizador Sintáctico

1.1. Gramática

$Y \rightarrow P$	$V \rightarrow - \text{id}$
$P \rightarrow B P$	$V \rightarrow \text{id}$
$P \rightarrow F P$	$V \rightarrow (E)$
$P \rightarrow \lambda$	$V \rightarrow \text{id} (L)$
$B \rightarrow \text{if} (E) S$	$V \rightarrow \text{ent}$
$B \rightarrow \text{let } T \text{ id} ;$	$V \rightarrow \text{cad}$
$B \rightarrow S$	$V \rightarrow \text{log}$
$B \rightarrow \text{for} (D ; E ; Z) C$	$X \rightarrow E$
$S \rightarrow \text{id} = E ;$	$X \rightarrow \lambda$
$S \rightarrow \text{id} (L) ;$	$L \rightarrow E Q$
$S \rightarrow \text{alert} (E) ;$	$L \rightarrow \lambda$
$S \rightarrow \text{input} (\text{id}) ;$	$Q \rightarrow , E Q$
$S \rightarrow \text{return } X ;$	$Q \rightarrow \lambda$
$T \rightarrow \text{number}$	$D \rightarrow \text{id} = E$
$T \rightarrow \text{boolean}$	$D \rightarrow \lambda$
$T \rightarrow \text{string}$	$Z \rightarrow \text{id} = E$
$F \rightarrow F1 F2 F3$	$Z \rightarrow - \text{id} R$
$F1 \rightarrow \text{function } H \text{ id}$	$Z \rightarrow \lambda$
$F2 \rightarrow (A)$	$H \rightarrow T$
$F3 \rightarrow C$	$H \rightarrow \lambda$
$E \rightarrow E \&\& R$	$A \rightarrow T \text{id} K$
$E \rightarrow R$	$A \rightarrow \lambda$
$R \rightarrow R == U$	$K \rightarrow , T \text{id} K$
$R \rightarrow U$	$K \rightarrow \lambda$
$U \rightarrow U - V$	$C \rightarrow B C$
$U \rightarrow U \cdot V$	$C \rightarrow \lambda$

1.2. Tabla LR

[illegible]

[illegible]

Como puede observarse en la tabla, esta gramática es adecuada para este tipo de Analizador sintáctico, puesto que no se produce ningún tipo de conflicto.

2. Anexo

A continuación se presentarán los casos de prueba para comprobar el funcionamiento de la gramática de la sección 1.1.

2.1. Casos de prueba correctos

2.1.1. Caso de prueba 1

Programa introducido:

```
function boolean bisiesto (number a)
{
    return (a - 4 == 0 && a - 100 == 0 && a - 400 == 0);
}
```

Parse:

Ascendente 15 45 18 14 50 47 19 28 26 31 25 24 31 26 23 22 28 26 31 25 24 31 26
23 21 28 26 31 25 24 31 26 23 21 29 26 24 22 34 13 7 52 51 20 17 4 3 1

VASt:

Arbol resultante de esta prueba

2.1.2. Caso de prueba 2

Programa introducido:

```
/* prueba correcta */
let number entero;
let string cadena;
let boolean logico;
let number abcdefghij_____;
function FuncionNumero ()
{
    let number i;
    let number j;

    logico = i == j;
}
```


Parse:

Ascendente 14 6 16 6 15 6 14 6 46 18 48 19 14 6 14 6 28 26 24 28 26 23 22 9 7 52
51 51 51 20 17 4 3 2 2 2 1

VASt:

Arbol resultante de esta prueba

2.1.3. Caso de prueba 3

Programa introducido:

```
function FuncionSentencia (number b, boolean z)
{
  for (b=1;z; )
  {
    alert (88);
  }
}
function Funcion (number x, boolean b)
{
  if (x == 0)FuncionSentencia(x,b);
  alert
    (cadena);return;
}
```

Parse:

Ascendente 46 18 14 15 50 49 47 19 31 26 24 22 40 28 26 24 22 44 31 26 24 22 11
7 52 51 8 52 51 20 17 46 18 14 15 50 49 47 19 28 26 24 31 26 23 22 28 26 24 22 28
26 24 22 39 38 36 10 5 28 26 24 22 11 7 35 13 7 52 51 51 51 20 17 4 3 3 1

VASt:

Arbol resultante de esta prueba

2.2. Casos de prueba incorrectos

2.2.1. Caso de prueba 1

Programa introducido:

```
function FuncionSentencia (number b, boolean z)
{
  for (b=1;z; ))
  {
    alert (88);
  }
}
```

Parse:

Ascendente 46 18 14 15 50 49 47 19 31 26 24 22 40 28 26 24 22 44

Error sintáctico:

Token ilegal CEPAREN en la linea 3

2.2.2. Caso de prueba 2

Programa introducido:

```
let number entero;
let string cadena;
let boolean logico;
let number abcdefghij_____;
function FuncionNumero ()
{
  let number i;
  let number j;

  logico = i = j;
}
function string FuncionRetorno (string hola, number x, boolean z)
{
  let number i;

  input (i);
  alert (hola);
  return string;
}
function FuncionSentencia (number b, boolean z)
```

```

{
  for (b=1;z; )
  {
    alert (88);
  }
}
function Funcion (number numbers, number x, boolean b)
{
  if (x == 0)FuncionSentencia(x,b);
  alert
    (cadena);return;
}
let number iii;
let boolean bbb;
/* comentario
multi
linea ***/
input(iii);
alert (iii) ;

iii = iii;
bbb = bbb;

Funcion
( iii,
  bbb );

```

Parse:

Ascendente 14 6 16 6 15 6 14 6 46 18 48 19 14 6 14 6

Error sintáctico:

Token ilegal OPASIG en la linea 11

2.2.3. Caso de prueba 3

Programa introducido:

```

let number iii;
let boolean bbb;
/* comentario
multi
linea ***/
input(iii);

```

```
alert (iii) ;

iii = iii function H (no&7/);
bbb = bbb;

Funcion
( iii,
  bbb );
```

Parse:

Ascendente 14 6 15 6 12 7 28 26 24 22 11 7

Error sintáctico:

Token ilegal FUNCTION en la linea 9