2022

PRÁCTICA PROCESADORES DEL LENGUAJE

GRUPO 13

INTEGRANTES:

- Mario López Estaire
- Andrés Bravo Francos
- Grettell Umpierrez Sardiñas

Sentencias: Sentencia de Selección Múltiple (switch-case)
Técnicas de Análisis Sintáctico: Descendente con tablas
Operadores especiales: Asignación con multiplicación (*=)
Comentarios: Comentario de bloque (/**/)
Cadenas: Con comillas dobles (“ ”)

ANALIZADOR SINTÁCTICO

GRAMÁTICA

```
// Axioma general
```

1. $\mathrm{A} \rightarrow \mathrm{B}\ \mathrm{A}$
2. $\mathrm{A} \rightarrow FA$
3. $\mathrm{A} \rightarrow eof$

```
// Sentencias compuestas
```

4. $B \rightarrow if\ (\ E\ )\ S$
5. $B \rightarrow let\ id\ T\ N$
6. $B \rightarrow S$
7. $B \rightarrow switch\ (U)\ \{\ Z\ \}$

```
// Sentencias simples
```

8. $S \rightarrow id\ S';$
9. $S \rightarrow print\ \ E;$
10. $S \rightarrow input\ id;$
11. $S \rightarrow return\ X;$
12. $S' \rightarrow = E$
13. $S' \rightarrow * = U$
14. $S' \rightarrow (\ L\ )$

```
// Expresiones
```

15. $E \rightarrow R\ E'$
16. $E' \rightarrow \lambda$
17. $E' \rightarrow \&\&\ R\ E'$ `// Operaciones lógicas`
18. $R \rightarrow U\ R'$
19. $R' \rightarrow \lambda$

20. $R' \rightarrow == \ U \ R'$ // Operaciones relacionales

21. $U \ \rightarrow \ V \ U'$

22. $U' \rightarrow \lambda$

23. $U' \rightarrow + V \ U'$ // Operaciones aritméticas (suma)

24. $V \rightarrow P \ V'$

25. $V' \rightarrow \lambda$

26. $V' \rightarrow * \ P \ V'$ // Operaciones aritméticas (producto)

// Operandos

27. $P \rightarrow id \ P'$

28. $P \rightarrow (\ E \ )$

29. $P \rightarrow cteEntera$

30. $P \rightarrow cadena$

31. $P \rightarrow true$

32. $P \rightarrow false$

33. $P' \rightarrow \ \lambda$

34. $P' \rightarrow (\ L \ )$

// Argumentos de función

35. $L \rightarrow E \ Q$

36. $L \rightarrow \lambda$

37. $Q \rightarrow , E \ Q$

38. $Q \rightarrow \lambda$

// Valor de retorno

39. $X \rightarrow E$

40. $X \rightarrow \ \lambda$

// Tipos de variables

41. T $\rightarrow$ int

42. $T \rightarrow$ boolean

43. $T \rightarrow string$

// Declaración de funciones

44. F $\rightarrow$ function id H (D) { C }

45. H $\rightarrow$ T

46. H $\rightarrow \lambda$

47. D → T id K
48. $D \rightarrow \lambda$
49. K → , T id K
50. $K \rightarrow \lambda$
51. C → B C
52. $C \rightarrow \lambda$

// Inicialización de identificadores

53. $N \rightarrow ;$
54. $N \rightarrow = E ;$
55. $N \rightarrow * = E;$

// Cuerpo de switch

56. $Z \rightarrow case\ cteEntera : O\ Z$
57. $Z \rightarrow default : O$
58. $O \rightarrow B\ O'$
59. $O \rightarrow break;$
60. $O' \rightarrow B\ O'$
61. $O' \rightarrow break;$
62. $O' \rightarrow \lambda$

FIRST Y FOLLOW

	FIRST	FOLLOW
A	if | let | switch | id | print | input | return | function | eof	eof
B	if | let | switch | id | print | input | return	if | let | switch | id | print | input | return | function | eof | } | break | case | default
S	id | print | input | return	if | let | switch | id | print | input | return | function | eof | } | break | case | default
S'	= | *= | (	;
E	id | (| cteEntera | cadena | true | false	) | ; | ,
E'	&& | λ	) | ; | ,

R	id | (| cteEntera | cadena | true | false	&& |) | ; | ,
R'	== | λ	&& |) | ; | ,
U	id | (| cteEntera | cadena | true | false	== | && |) | ; | ,
U'	+ | λ	== | && |) | ; | ,
V	id | (| cteEntera | cadena | true | false	+ | == | && |) | ; | ,
V'	* | λ	+ | == | && |) | ; | ,
P	id | (| cteEntera | cadena | true | false	* | + | == | && |) | ; | ,
P'	(| λ	* | + | == | && |) | ; | ,
L	id | (| cteEntera | cadena | true | false | λ	)
Q	, | λ	)
X	id | (| cteEntera | cadena | true | false | λ	;
T	int | boolean | string	id | (| ; | = | *=
F	function	if | let | switch | id | print | input | return | function | eof
H	int | boolean | string | λ	(
D	int | boolean | string | λ	)
K	, | λ	)
C	if | let | switch | id | print | input | return | λ	}
N	; | = | *=	if | let | switch | id | print | input | return | function | eof | } | break | case | default
Z	case | default	}
O	if | let | switch | id | print | input | return | break	case | default | }
O'	if | let | switch | id | print | input | return | break | λ	case | default | }

Justificación

Verificación condición LL

Las reglas de A cumplen la condición LL porque:

- $First(BA) \cap First(FA) = \emptyset$ porque $\{ if \mid let \mid switch \mid id \mid print \mid input \mid return \} \cap \{ \text{function} \} = \emptyset$
- $First(FA) \cap First(eof) = \emptyset$ porque $\{ function \} \cap \{ eof \} = \emptyset$
- $First(BA) \cap First(eof) = \emptyset$ porque $\{ if \mid let \mid switch \mid id \mid print \mid input \mid return \} \cap \{ \text{eof} \} = \emptyset$
- No hay regla $\boldsymbol{\lambda}$ por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de B cumplen la condición LL porque:

- $First(if\ (E)\ S) \cap First(let\ id\ T\ N) = \emptyset$ porque $\{ if \} \cap \{ let \} = \emptyset$
- $First(if\ (E)\ S) \cap First(S) = \emptyset$ porque $\{ if \} \cap \{ id \mid print \mid input \mid return \} = \emptyset$
- $First(if\ (E)\ S) \cap First(switch\ (U)\ \{ Z \}) = \emptyset$ porque $\{ if \} \cap \{ switch \} = \emptyset$
- $First(let\ id\ T\ N) \cap First(S) = \emptyset$ porque $\{ let \} \cap \{ id \mid print \mid input \mid return \} = \emptyset$
- $First(let\ id\ T\ N) \cap First(\ switch\ (U)\ \{ Z \}) = \emptyset$ porque $\{ let \} \cap \{ switch \} = \emptyset$
- $First(S) \cap First(\ switch\ (U)\ \{ Z \}) = \emptyset$ porque $\{ id \mid print \mid input \mid return \} \cap \{ switch \} = \emptyset$
- No hay regla $\boldsymbol{\lambda}$ por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de S cumplen la condición LL porque:

- $First(id\ S';) \cap First(print\ E\ ;) = \emptyset$ porque $\{ id \} \cap \{ print \} = \emptyset$
- $First(id\ S';) \cap First(\ input\ id\ ;) = \emptyset$ porque $\{ id \} \cap \{ input \} = \emptyset$
- $First(id\ S';) \cap First(return\ X\ ;) = \emptyset$ porque $\{ id \} \cap \{ return \} = \emptyset$
- $First(print\ E;) \cap First(input\ id\ ;) = \emptyset$ porque $\{ print \} \cap \{ input \} = \emptyset$
- $First(print\ E;) \cap First(\ return\ X\ ;) = \emptyset$ porque $\{ print \} \cap \{ return \} = \emptyset$
- $First(\ input\ id\ ;) \cap First(\ return\ X\ ;) = \emptyset$ porque $\{ input \} \cap \{ return \} = \emptyset$
- No hay regla $\boldsymbol{\lambda}$ por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de S' cumplen la condición LL porque:

- $First(= E) \cap First(*= U) = \emptyset$ porque $\{ = \} \cap \{ *= \} = \emptyset$
- $First(= E) \cap First(\ (\ L\ )\ ) = \emptyset$ porque $\{ = \} \cap \{ (\} = \emptyset$
- $First(*= U) \cap First(\ (\ L\ )\ ) = \emptyset$ porque $\{ *= \} \cap \{ (\} = \emptyset$
- No hay regla $\boldsymbol{\lambda}$ por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de E cumplen la condición LL porque:

- Solamente tiene una regla por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de E' cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es $\boldsymbol{\lambda}$ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es $\boldsymbol{\lambda}$
    - $First(\ \&\&\ R\ E')\cap Follow(\ E'\ )=\emptyset$ porque $\{\ \&\&\}\ \cap\{\ )\mid;\mid,\}=\emptyset$ por tanto se cumple la condición LL.

Las reglas de R cumplen la condición LL porque:

- Solamente tiene una regla por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de R' cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es $\boldsymbol{\lambda}$ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es $\boldsymbol{\lambda}$
    - $First(==\ U\ R')\cap Follow(\ R'\ )=\emptyset$ porque $\{==\}\ \cap\{\ \&\&\mid\ )\mid;\mid,\}=\emptyset$ por tanto se cumple la condición LL.

Las reglas de U cumplen la condición LL porque:

- Solamente tiene una regla por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de U' cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es $\boldsymbol{\lambda}$ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es $\boldsymbol{\lambda}$
    - $First(+\ V\ U')\cap Follow(\ U'\ )=\emptyset$ porque $\{+\}\ \cap\{==\mid\ \&\&\mid\ )\mid;\mid,\}=\emptyset$ por tanto se cumple la condición LL.

Las reglas de V cumplen la condición LL porque:

- Solamente tiene una regla por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de V' cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es $\boldsymbol{\lambda}$ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es $\boldsymbol{\lambda}$
    - $First(*\ P\ V')\cap Follow(\ V'\ )=\emptyset$ porque $\{*\}\ \cap\{+\mid\ ==\mid\ \&\&\mid\ )\mid;\mid,\}=\emptyset$ por tanto se cumple la condición LL.

Las reglas de P cumplen la condición LL porque:

- $First(\ id\ P')\cap First\big((\ E\ )\big)=\emptyset$ porque $\{\ id\ \}\cap\{\ (\ \}=\emptyset$
- $First(\ id\ P'\ )\cap First(\ cteEntera\ )=\emptyset$ porque $\{\ id\ \}\ \cap\{\ cteEntera\ \}=\emptyset$

- $First(id\ P') \cap First(\ true\ ) = \emptyset$ porque $\{\ id\ \} \cap \{\ true\ \} = \emptyset$
- $First(id\ P') \cap First(\ false\ ) = \emptyset$ porque $\{\ id\ \} \cap \{\ false\ \} = \emptyset$
- No hay regla $\boldsymbol{\lambda}$ por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de P' cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es $\boldsymbol{\lambda}$ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es $\boldsymbol{\lambda}$
    - $First\big((\ L)\big) \cap Follow(\ P'\ ) = \emptyset$ porque $\{\ (\ \} \cap \{\ * \mid + \mid == \mid \&\& \mid\ ) \mid ; \mid , \} = \emptyset$ por tanto se cumple la condición LL.

Las reglas de L cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es $\boldsymbol{\lambda}$ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es $\boldsymbol{\lambda}$
    - $First(E\ Q) \cap Follow(\ L\ ) = \emptyset$ porque $\{\ id\ \mid\ (\ \mid\ cteEntera\ \mid cadena \mid true\ \mid\ false\ \} \cap \{\ )\ \} = \emptyset$ por tanto se cumple la condición LL.

Las reglas de Q cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es $\boldsymbol{\lambda}$ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es $\boldsymbol{\lambda}$
    - $First(\ ,E\ Q\ ) \cap Follow(\ Q\ ) = \emptyset$ porque $\{\ ,\ \} \cap \{\ )\ \} = \emptyset$ por tanto se cumple la condición LL.

Las reglas de X cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es $\boldsymbol{\lambda}$ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es $\boldsymbol{\lambda}$
    - $First(\ E\ ) \cap Follow(\ X\ ) = \emptyset$ porque $\{\ id\ \mid\ (\ \mid\ cteEntera\ \mid cadena \mid true\ \mid\ false\ \} \cap \{\ ;\ \} = \emptyset$ por tanto se cumple la condición LL.

Las reglas de T cumplen la condición LL porque:

- $First(\ int\ ) \cap First(\ boolean\ ) = \emptyset$ porque $\{\ int\ \} \cap \{\ boolean\ \} = \emptyset$
- $First(\ int\ ) \cap First(\ string\ ) = \emptyset$ porque $\{\ int\ \} \cap \{\ string\ \} = \emptyset$
- $First(\ boolean\ ) \cap First(\ string\ ) = \emptyset$ porque $\{\ boolean\ \} \cap \{\ string\ \} = \emptyset$
- No hay regla $\boldsymbol{\lambda}$ por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de T cumplen la condición LL porque:

- $First(\ int\ ) \cap First(\ boolean\ ) = \emptyset$ porque $\{\ int\ \} \cap \{\ boolean\ \} = \emptyset$
- $First(\ int\ ) \cap First(\ string\ ) = \emptyset$ porque $\{\ int\ \} \cap \{\ string\ \} = \emptyset$
- $First(\ boolean\ ) \cap First(\ string\ ) = \emptyset$ porque $\{\ boolean\ \} \cap \{\ string\ \} = \emptyset$

- No hay regla λ por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de F cumplen la condición LL porque:

- Solamente tiene una regla por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de H cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es λ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es λ
    - $First(\ T\ ) \cap Follow(\ H\ ) = \emptyset$ porque $\{\ int \mid boolean \mid string\ \} \cap \{\ ;\ \} = \emptyset$ por tanto se cumple la condición LL.

Las reglas de D cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es λ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es λ
    - $First(\ T\ \ id\ K\ ) \cap Follow(\ D\ ) = \emptyset$ porque $\{\ int \mid boolean \mid string\ \} \cap \{\ )\ \} = \emptyset$ por tanto se cumple la condición LL.

Las reglas de K cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es λ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es λ
    - $First(\ ,T\ \ id\ K\ ) \cap Follow(\ K\ ) = \emptyset$ porque $\{\ ,\ \} \cap \{\ )\ \} = \emptyset$ por tanto se cumple la condición LL.

Las reglas de C cumplen la condición LL porque:

- Solamente hay dos reglas y una de ellas es λ por tanto no hay que comprobar si los first tienen símbolos terminales en común
- Como una de las reglas es λ
    - $First(\ B\ C\ ) \cap Follow(\ C\ ) = \emptyset$ porque $\{\ if \mid let \mid switch \mid id \mid print \mid input\ \ \} \cap \{\ \}\ \} = \emptyset$ por tanto se cumple la condición LL.

Las reglas de N cumplen la condición LL porque:

- $First(\ ;\ ) \cap First(= E\ ) = \emptyset$ porque $\{\ ;\ \} \cap \{=\} = \emptyset$
- $First(\ ;\ ) \cap First(*= E\ ) = \emptyset$ porque $\{\ ;\ \} \cap \{\ *=\ \} = \emptyset$
- $First(= E\ ) \cap First(\ *= E\ ) = \emptyset$ porque $\{\ =\ \} \cap \{\ *=\ \} = \emptyset$
- No hay regla λ por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de Z cumplen la condición LL porque:

- $First(\ case\ cteEntera : O\ Z) \cap First(default : O\ ) = \emptyset$ porque $\{\ case\ \} \cap \{\ default\ \ \} = \emptyset$
- No hay regla λ por tanto no se necesitan más comprobaciones para determinar que se cumple la condición

LL

Las reglas de O cumplen la condición LL porque:

- $First(\ B\ O'\ ) \cap First(\ break\ ;\ ) = \emptyset$ porque $\{\ if\ |\ let\ |\ switch\ |\ id\ |\ print\ |\ input\ \ \} \cap \{\ break\ \} = \emptyset$
- No hay regla λ por tanto no se necesitan más comprobaciones para determinar que se cumple la condición LL

Las reglas de O' cumplen la condición LL porque:

- $First(\ B\ O'\ ) \cap First(\ break\ ;\ ) = \emptyset$ porque $\{\ if\ |\ let\ |\ switch\ |\ id\ |\ print\ |\ input\ \ \} \cap \{\ break\ \} = \emptyset$
- Como una de las reglas es λ
  - $First(\ B\ O'\ ) \cap Follow(\ O'\ ) = \emptyset$ porque $\{\ if\ |\ let\ |\ switch\ |\ id\ |\ print\ |\ input\ \ \} \cap \{\ case\ |\ default\ |\ \}\} = \emptyset$
  - $First(\ break;\ ) \cap Follow(\ O'\ ) = \emptyset$ porque $\{\ break\ \} \cap \{\ case\ |\ default\ |\ \}\} = \emptyset$ por tanto se cumple la condición LL.

La gramática obtenida para el Analizador Sintáctico Descendente LL(1) con Tablas es LL(1) porque:

- Para cada no terminal para el que haya más de una regla de producción, dichas reglas no derivan un mismo terminal (la gramática está factorizada).
- Cumple la condición LL
- No existe recursividad por la izquierda

TABLA A.S DESCENDENTE (LL)

	if	let	switch	id	print	input	return	function	break	cteEntera	cadena	true	false	==	&&	;	,	(	)	{	}	+	*	=	*=	int	boolean	string	case	default	:	eof
A	1	1	1	1	1	1	1	2	101	101	101	101	101	101	101	101	101	101	101	101	101	101	101	101	101	101	101	101	101	101	101	3
B	4	5	7	6	6	6	6	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102
S	103	103	103	8	9	10	11	103	103	103	103	103	103	103	103	103	103	103	103	103	103	103	103	103	103	103	103	103	103	103	103	103
S'	104	104	104	104	104	104	104	104	104	104	104	104	104	104	104	104	104	14	104	104	104	104	104	12	13	104	104	104	104	104	104	104
E	105	105	105	15	105	105	105	105	105	15	15	15	15	105	105	105	105	15	105	105	105	105	105	105	105	105	105	105	105	105	105	105
E'	106	106	106	106	106	106	106	106	106	106	106	106	106	17	16	16	106	16	106	106	106	106	106	106	106	106	106	106	106	106	106	106
R	107	107	107	18	107	107	107	107	107	18	18	18	18	107	107	107	107	18	107	107	107	107	107	107	107	107	107	107	107	107	107	107
R'	108	108	108	108	108	108	108	108	108	108	108	108	108	20	19	19	19	108	19	108	108	108	108	108	108	108	108	108	108	108	108	108
U	109	109	109	21	109	109	109	109	109	21	21	21	21	109	109	109	109	21	109	109	109	109	109	109	109	109	109	109	109	109	109	109
U'	110	110	110	110	110	110	110	110	110	110	110	110	110	22	22	22	22	110	22	110	110	23	110	110	110	110	110	110	110	110	110	110
V	111	111	111	24	111	111	111	111	111	24	24	24	24	111	111	111	111	24	111	111	111	111	111	111	111	111	111	111	111	111	111	111
V'	112	112	112	112	112	112	112	112	112	112	112	112	112	25	25	25	25	112	25	112	112	25	26	112	112	112	112	112	112	112	112	112
P	113	113	113	27	113	113	113	113	113	29	30	31	32	113	113	113	113	28	113	113	113	113	113	113	113	113	113	113	113	113	113	113
P'	114	114	114	114	114	114	114	114	114	114	114	114	114	33	33	33	33	34	33	114	114	33	33	114	114	114	114	114	114	114	114	114
L	115	115	115	35	115	115	115	115	115	35	35	35	35	115	115	115	115	35	36	115	115	115	115	115	115	115	115	115	115	115	115	115
Q	116	116	116	116	116	116	116	116	116	116	116	116	116	116	116	116	37	116	38	116	116	116	116	116	116	116	116	116	116	116	116	116
X	117	117	117	39	117	117	117	117	117	39	39	39	39	117	117	40	117	39	117	117	117	117	117	117	117	117	117	117	117	117	117	117
T	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118	118	41	42	43	118	118	118	118
F	119	119	119	119	119	119	119	44	119	119	119	119	119	119	119	119	119	119	119	119	119	119	119	119	119	119	119	119	119	119	119	119
H	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	46	120	120	120	120	120	120	120	45	45	45	120	120	120	120
D	121	121	121	121	121	121	121	121	121	121	121	121	121	121	121	121	121	121	48	121	121	121	121	121	121	47	47	47	121	121	121	121
K	122	122	122	122	122	122	122	122	122	122	122	122	122	122	122	122	49	122	50	122	122	122	122	122	122	122	122	122	122	122	122	122
C	51	51	51	51	51	51	51	123	123	123	123	123	123	123	123	123	123	123	123	123	123	52	123	123	123	123	123	123	123	123	123	123
N	124	124	124	124	124	124	124	124	124	124	124	124	124	124	124	53	124	124	124	124	124	124	124	54	55	124	124	124	124	124	124	124
Z	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	125	56	57	125	125
O	58	58	58	58	58	58	58	126	59	126	126	126	126	126	126	126	126	126	126	126	126	126	126	126	126	126	126	126	126	126	126	126
O'	60	60	60	60	60	60	60	127	61	127	127	127	127	127	127	127	127	127	127	127	62	127	127	127	127	127	127	127	62	62	127	127

		Error Sintáctico

ERRORES

Sintácticos

Código 100: Token *siguiente_token* no esperado. Se esperaba *cima_pila*.

Código 101: El token *siguiente_token* no pertenece al primer elemento de una sentencia válida, una función o es fin de fichero.

Código 102: El token *siguiente_token* no pertenece al primer elemento de una sentencia válida.

Código 103: El token *siguiente_token* no pertenece al primer elemento de una sentencia simple válida.

Código 104: El token *siguiente_token* no es una asignación válida o la llamada a una función.

Código 105: El token *siguiente_token* no pertenece a una expresión válida.

Código 106: El token *siguiente_token* no pertenece a una expresión lógica u otra expresión válida.

Código 107: El token *siguiente_token* no pertenece a una expresión válida.

Código 108: El token *siguiente_token* no pertenece a una expresión relacional u otra expresión válida.

Código 109: El token *siguiente_token* no pertenece a una expresión válida.

Código 110: El token *siguiente_token* no pertenece a una expresión aritmética de suma u otra expresión válida.

Código 111: El token *siguiente_token* no pertenece a una expresión válida.

Código 112: El token *siguiente_token* no pertenece a una expresión aritmética de multiplicación u otra expresión válida.

Código 113: El token *siguiente_token* no pertenece a una expresión válida.

Código 114: El token *siguiente_token* no es un identificador de variable o la llamada a una función

Código 115: El token *siguiente_token* no es un argumento de función válido porque no pertenece a una expresión válida

Código 116: El token *siguiente_token* no es un argumento de función válido porque no pertenece a una expresión válida

Código 117: El token *siguiente_token* no es un valor de retorno de función válido porque no pertenece a una expresión válida

Código 118: El token *siguiente_token* no es un tipo de variable válido

Código 119: El token *siguiente_token* no pertenece a una declaración válida de una función

Código 120: El token *siguiente_token* no es un tipo de valor de retorno de función válido

Código 121: El token *siguiente_token* no es una declaración válida de un argumento de una función

Código 122: El token *siguiente_token* no es una declaración válida de un argumento de una función

Código 123: El token *siguiente_token* no pertenece a una sentencia válida

Código 124: El token *siguiente_token* no es una asignación válida

Código 125: El token *siguiente_token* no es un cuerpo válido para la condicional múltiple switch

Código 126: El token *siguiente_token* no pertenece a una sentencia válida

Código 127: El token *siguiente_token* no pertenece a una sentencia válida

- *siguiente_token*: último token devuelto por el Analizador Léxico
- *cima_pila:* último token apilado dado el algoritmo del Analizador Sintáctico Descendente por Tablas.

ANEXOS

CORRECTOS

Caso de Prueba # 1

Código fuente

```
/*
    CASO DE PRUEBA #1
*/

print "Introduce el resultado de 6+6*2: ";
let a_1 int = 0;

input a_1;

let num int = 5;

if (a_1 == 18)
    print "Bien!!";
if (a_1 == 18)    print "Introduce un número para sumarle a 18: ";
if (a_1 == 18)    input num;


switch (a_1) {

    case 24:
        print ":( es 6*2 = 12 y luego 12 + 6 = 18 no 24";
        print "Introduce un número para sumarle a 24: ";
        input num;
        break;
    default:
        print "Vaya " + a_1 + " no es correcto.";
}


function Suma int (int a, int b) {

    j= a + b;
    return j;
    /* La función finaliza y devuelve el valor entero de la expresión */
}

print a_1 + " + " + num + " = ";
```

```
print  Suma(a_1,num);
```

Parse

Descendente

```
1    6    9    15   18   21   24   30   25   22   19   16   1    5
     41   54   15   18   21   24
29   25   22   19   16   1    6    10   1    5    41   54   15   18
     21   24   29   25   22   19
16   1    4    15   18   21   24   27   33   25   22   20   21   24
     29   25   22   19   16   9
15   18   21   24   30   25   22   19   16   1    4    15   18   21
     24   27   33   25   22   20
21   24   29   25   22   19   16   9    15   18   21   24   30   25
     22   19   16   1    4    15
18   21   24   27   33   25   22   20   21   24   29   25   22   19
     16   10   1    7    21   24
27   33   25   22   56   58   6    9    15   18   21   24   30   25
     22   19   16   60   6    9
15   18   21   24   30   25   22   19   16   60   6    10   61   57
     58   6    9    15   18   21
24   30   25   23   24   27   33   25   23   24   30   25   22   19
     16   62   2    44   45   41
47   41   49   41   50   51   6    8    12   15   18   21   24   27
     33   25   23   24   27   33
25   22   19   16   51   6    11   39   15   18   21   24   27   33
     25   22   19   16   52   1
6    9    15   18   21   24   27   33   25   23   24   30   25   23
     24   27   33   25   23   24
30   25   22   19   16   1    6    9    15   18   21   24   27   34
     35   15   18   21   24   27
33   25   22   19   16   37   15   18   21   24   27   33   25   22
     19   16   38   25   22   19
16   3
```

Árbol

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" 'http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd'>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1"/>
<title>Arbol</title>
<link rel="stylesheet" href="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/jquery.treeview.css" />
<link rel="stylesheet" href="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/screen.css" />
<script src="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/jquery.js" type="text/javascript"></script>
<script src="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/jquery.treeview.js" type="text/javascript"></script>
<link rel="stylesheet" type="text/css" href="CSS.css">
<style type="text/css">
```

```
.filetree span.Noterminales { color: Brown;}
.filetree span.terminales { color: Red;}
</style>
<script>
$(document).ready(function(){
$("#browser").treeview();
});
</script>
</head>
<body>
<h4>Árbol resultado de:</h4>
<h4>Gramática: E:\Grettell\Universidad\UPM\Segundo Curso\Primer Semestre\Procesadores de Lenguaje\Práctica\Proyecto\Visualizador Arboles Sintacticos\GramaticaAS.txt</h4>
<h4>Parse: E:\Grettell\Universidad\UPM\Segundo Curso\Primer Semestre\Procesadores de Lenguaje\Práctica\Proyecto\Visualizador Arboles Sintacticos\Casos de Prueba E2\Caso 1\parseC1.txt</h4>
<ul id="browser" class="filetree">

<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (9)</span>
<ul>
<li><span class="terminales">print</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (30)</span>
<ul>
<li><span class="terminales">cadena</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
```

```
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (5)</span>
<ul>
<li><span class="terminales">let</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">T (41)</span>
<ul>
<li><span class="terminales">int</span>
</li>
</ul>
</li>
<li><span class="Noterminales">N (54)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (29)</span>
<ul>
<li><span class="terminales">cteEntera</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
```

lambda

Up (22)

lambda

Rp (19)

lambda

Ep (16)

lambda

;

A (1)

B (6)

S (10)

input

id

;

A (1)

B (5)

```
<ul>
<li><span class="terminales">let</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">T (41)</span>
<ul>
<li><span class="terminales">int</span>
</li>
</ul>
</li>
<li><span class="Noterminales">N (54)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (29)</span>
<ul>
<li><span class="terminales">cteEntera</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
```

lambda

;

A (1)

B (4)

if

(

E (15)

R (18)

U (21)

V (24)

P (27)

id

Pp (33)

lambda

Vp (25)

lambda

Up (22)

lambda


```
</li>
<li><span class="Noterminales">Rp (20)</span>
<ul>
<li><span class="terminales">==</span>
</li>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (29)</span>
<ul>
<li><span class="terminales">cteEntera</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="Noterminales">S (9)</span>
<ul>
<li><span class="terminales">print</span>
</li>
```

E (15)

R (18)

U (21)

V (24)

P (30)

cadena

Vp (25)

lambda

Up (22)

lambda

Rp (19)

lambda

Ep (16)

lambda

;

A (1)

B (4)

```
<ul>
<li><span class="terminales">if</span>
</li>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (20)</span>
<ul>
<li><span class="terminales">==</span>
</li>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (29)</span>
<ul>
<li><span class="terminales">cteEntera</span>
</li>
</ul>
</li>
```

Vp (25)

lambda

Up (22)

lambda

Rp (19)

lambda

Ep (16)

lambda

)

S (9)

print

E (15)

R (18)

U (21)

V (24)

P (30)

cadena

Vp (25)

lambda

Up (22)

lambda

Rp (19)

lambda

Ep (16)

lambda

;

A (1)

B (4)

if

(

E (15)

R (18)

U (21)

V (24)

P (27)

id

Pp (33)

lambda

Vp (25)

lambda

Up (22)

lambda

Rp (20)

==

U (21)

V (24)

P (29)

cteEntera

Vp (25)

lambda

Up (22)

lambda


```
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="Noterminales">S (10)</span>
<ul>
<li><span class="terminales">input</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (7)</span>
<ul>
<li><span class="terminales">switch</span>
</li>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
```

```
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="terminales">{</span>
</li>
<li><span class="Noterminales">Z (56)</span>
<ul>
<li><span class="terminales">case</span>
</li>
<li><span class="terminales">cteEntera</span>
</li>
<li><span class="terminales">:</span>
</li>
<li><span class="Noterminales">O (58)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (9)</span>
<ul>
<li><span class="terminales">print</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (30)</span>
<ul>
<li><span class="terminales">cadena</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
```

```
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Op (60)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (9)</span>
<ul>
<li><span class="terminales">print</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (30)</span>
<ul>
<li><span class="terminales">cadena</span>
</li>
```

```
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Op (60)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (10)</span>
<ul>
<li><span class="terminales">input</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
```

```
</li>
<li><span class="Noterminales">Op (61)</span>
<ul>
<li><span class="terminales">break</span>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Z (57)</span>
<ul>
<li><span class="terminales">default</span>
</li>
<li><span class="terminales">:</span>
</li>
<li><span class="Noterminales">O (58)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (9)</span>
<ul>
<li><span class="terminales">print</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (30)</span>
<ul>
<li><span class="terminales">cadena</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (23)</span>
<ul>
<li><span class="terminales">+</span>
</li>
```

V (24)

P (27)

id

Pp (33)

lambda

Vp (25)

lambda

Up (23)

+

V (24)

P (30)

cadena

Vp (25)

lambda

Up (22)

lambda

Rp (19)

```
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Op (62)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">}</span>
</li>
</ul>
</li>
<li><span class="Noterminales">A (2)</span>
<ul>
<li><span class="Noterminales">F (44)</span>
<ul>
<li><span class="terminales">function</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">H (45)</span>
<ul>
<li><span class="Noterminales">T (41)</span>
<ul>
<li><span class="terminales">int</span>
</li>
</ul>
</li>
</ul>
</li>
```

```
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">D (47)</span>
<ul>
<li><span class="Noterminales">T (41)</span>
<ul>
<li><span class="terminales">int</span>
</li>
</ul>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">K (49)</span>
<ul>
<li><span class="terminales">,</span>
</li>
<li><span class="Noterminales">T (41)</span>
<ul>
<li><span class="terminales">int</span>
</li>
</ul>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">K (50)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="terminales">{</span>
</li>
<li><span class="Noterminales">C (51)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Sp (12)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
```

```
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (23)</span>
<ul>
<li><span class="terminales">+</span>
</li>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
```

```
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">C (51)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (11)</span>
<ul>
<li><span class="terminales">return</span>
</li>
<li><span class="Noterminales">X (39)</span>
<ul>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
```

```
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">C (52)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">}</span>
</li>
```

```
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (9)</span>
<ul>
<li><span class="terminales">print</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (23)</span>
<ul>
<li><span class="terminales">+</span>
</li>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (30)</span>
<ul>
<li><span class="terminales">cadena</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
```

```
</li>
</ul>
</li>
<li><span class="Noterminales">Up (23)</span>
<ul>
<li><span class="terminales">+</span>
</li>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (23)</span>
<ul>
<li><span class="terminales">+</span>
</li>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (30)</span>
<ul>
<li><span class="terminales">cadena</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
```

```
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (9)</span>
<ul>
<li><span class="terminales">print</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (34)</span>
<ul>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">L (35)</span>
```

```
<ul>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Q (37)</span>
<ul>
```

,

E (15)

R (18)

U (21)

V (24)

P (27)

id

Pp (33)

lambda

Vp (25)

lambda

Up (22)

lambda

Rp (19)

lambda

Ep (16)

lambda

Q (38)

lambda

)

Vp (25)

lambda

Up (22)

lambda

Rp (19)

lambda

Ep (16)

lambda

;

A (3)


```
<li><span class="terminales">eof</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</body>
</html>
```

Caso de Prueba # 2

Código fuente

```
/*
    CASO DE PRUEBA #2
*/

let unaCadena64 string = "Si la vida te da limones haz limonada. Quieres limones
?????????"

let _un_num_1 int = 32767;  /* a una unidad del error */

let a int = _un_num_1;

let bool boolean = true;

if (a == _un_num_1 &&bool) bool = false;



function PrintRecursivo boolean (string cadena) {
```

```
    print cadena;

    if(bool == false) bool = true;
    if(bool == false) PrintRecursivo(cadena);

    return true;
}

a *= a;
_un_num_1 = a * a;

switch(a){
    case _un_num_1:
        PrintRecursivo(unaCadena64);
        break;
    default:
        PrintRecursivo(unaCadena64);
}
```

Parse

Descendente

```
1    5    43    54    15    18    21    24    30    25    22    19    16    1    5
     41   54    15    18    21
24   29   25    22    19    16    1     5     41    54    15    18    21    24
     27   33    25    22    19    16
1    5    42    54    15    18    21    24    31    25    22    19    16    1    4
     15   18    21    24    27
33   25   22    20    21    24    27    33    25    22    19    17    18    21
     24   27    33    25    22    19
16   8    12    15    18    21    24    32    25    22    19    16    2     44
     45   42    47    43    50    51
6    9    15    18    21    24    27    33    25    22    19    16    51    4
     15   18    21    24    27    33
25   22   20    21    24    32    25    22    19    16    8     12    15    18
     21   24    31    25    22    19
16   51   4     15    18    21    24    27    33    25    22    20    21    24
     32   25    22    19    16    8
14   35   15    18    21    24    27    33    25    22    19    16    38    51    6
     11   39    15    18    21
24   31   25    22    19    16    52    1     6     8     13    21    24    27
     33   25    22    1     6     8
12   15   18    21    24    27    33    26    27    33    25    22    19    16    1
     7    21    24    27    33
25   22   56    58    6     8     14    35    15    18    21    24    27    33
     25   22    19    16    38    61
```

57 58 6 8 14 35 15 18 21 24 27 33 25 22
19 16 38 62 3

Árbol

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" 'http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd'>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1"/>
<title>Arbol</title>
<link rel="stylesheet" href="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/jquery.treeview.css" />
<link rel="stylesheet" href="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/screen.css" />
<script src="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/jquery.js" type="text/javascript"></script>
<script src="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/jquery.treeview.js" type="text/javascript"></script>
<link rel="stylesheet" type="text/css" href="CSS.css">
<style type="text/css">
.filetree span.Noterminales { color: Brown;}
.filetree span.terminales { color: Red;}
</style>
<script>
$(document).ready(function(){
$("#browser").treeview();
});
</script>
</head>
<body>
<h4>Árbol resultado de:</h4>
<h4>Gramática: E:\Grettell\Universidad\UPM\Segundo Curso\Primer Semestre\Procesadores de Lenguaje\Práctica\Proyecto\Visualizador Arboles Sintacticos\GramaticaAS.txt</h4>
<h4>Parse: E:\Grettell\Universidad\UPM\Segundo Curso\Primer Semestre\Procesadores de Lenguaje\Práctica\Proyecto\Visualizador Arboles Sintacticos\Casos de Prueba E2\Caso 2\parseC2.txt</h4>
<ul id="browser" class="filetree">

<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (5)</span>
<ul>
<li><span class="terminales">let</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">T (43)</span>
<ul>
<li><span class="terminales">string</span>
</li>
</ul>
</li>
<li><span class="Noterminales">N (54)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
```

```
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (30)</span>
<ul>
<li><span class="terminales">cadena</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (5)</span>
<ul>
<li><span class="terminales">let</span>
</li>
```

```
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">T (41)</span>
<ul>
<li><span class="terminales">int</span>
</li>
</ul>
</li>
<li><span class="Noterminales">N (54)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (29)</span>
<ul>
<li><span class="terminales">cteEntera</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
```

```
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (5)</span>
<ul>
<li><span class="terminales">let</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">T (41)</span>
<ul>
<li><span class="terminales">int</span>
</li>
</ul>
</li>
<li><span class="Noterminales">N (54)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
```

Up (22)

lambda

Rp (19)

lambda

Ep (16)

lambda

;

A (1)

B (5)

let

id

T (42)

boolean

N (54)

=

E (15)

R (18)

U (21)

V (24)

```
<ul>
<li><span class="Noterminales">P (31)</span>
<ul>
<li><span class="terminales">true</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (4)</span>
<ul>
<li><span class="terminales">if</span>
</li>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
```

R (18)

U (21)

V (24)

P (27)

id

Pp (33)

lambda

Vp (25)

lambda

Up (22)

lambda

Rp (20)

==

U (21)

V (24)

P (27)

id

Pp (33)

lambda

Vp (25)

```
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (17)</span>
<ul>
<li><span class="terminales">&&</span>
</li>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
```

Up (22)

lambda

Rp (19)

lambda

Ep (16)

lambda

)

S (8)

id

Sp (12)

=

E (15)

R (18)

U (21)

V (24)

P (32)

false

Vp (25)

lambda


```
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (2)</span>
<ul>
<li><span class="Noterminales">F (44)</span>
<ul>
<li><span class="terminales">function</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">H (45)</span>
<ul>
<li><span class="Noterminales">T (42)</span>
<ul>
<li><span class="terminales">boolean</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">(</span>
</li>
```

```
<li><span class="Noterminales">D (47)</span>
<ul>
<li><span class="Noterminales">T (43)</span>
<ul>
<li><span class="terminales">string</span>
</li>
</ul>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">K (50)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="terminales">{</span>
</li>
<li><span class="Noterminales">C (51)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (9)</span>
<ul>
<li><span class="terminales">print</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
```

```
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">C (51)</span>
<ul>
<li><span class="Noterminales">B (4)</span>
<ul>
<li><span class="terminales">if</span>
</li>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
```

lambda

Vp (25)

lambda

Up (22)

lambda

Rp (20)

==

U (21)

V (24)

P (32)

false

Vp (25)

lambda

Up (22)

lambda

Rp (19)

lambda

```
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Sp (12)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (31)</span>
<ul>
<li><span class="terminales">true</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
```

```
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">C (51)</span>
<ul>
<li><span class="Noterminales">B (4)</span>
<ul>
<li><span class="terminales">if</span>
</li>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
```

```
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (20)</span>
<ul>
<li><span class="terminales">==</span>
</li>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (32)</span>
<ul>
<li><span class="terminales">false</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
```

```
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Sp (14)</span>
<ul>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">L (35)</span>
<ul>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
```

```
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Q (38)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">C (51)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (11)</span>
<ul>
<li><span class="terminales">return</span>
</li>
<li><span class="Noterminales">X (39)</span>
<ul>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (31)</span>
<ul>
<li><span class="terminales">true</span>
</li>
```

```
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">C (52)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
```

```
</li>
<li><span class="terminales">}</span>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Sp (13)</span>
<ul>
<li><span class="terminales">*=</span>
</li>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
```

```
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Sp (12)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (26)</span>
<ul>
<li><span class="terminales">*</span>
</li>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
```

```
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (7)</span>
<ul>
<li><span class="terminales">switch</span>
</li>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
```

lambda

Vp (25)

lambda

Up (22)

lambda

)

{

Z (56)

case

cteEntera

:

O (58)

B (6)

S (8)

id

Sp (14)

(

L (35)

E (15)

R (18)

U (21)

V (24)

```
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Q (38)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
```

```
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Op (61)</span>
<ul>
<li><span class="terminales">break</span>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Z (57)</span>
<ul>
<li><span class="terminales">default</span>
</li>
<li><span class="terminales">:</span>
</li>
<li><span class="Noterminales">O (58)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Sp (14)</span>
<ul>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">L (35)</span>
<ul>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
```

```
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Q (38)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Op (62)</span>
<ul>
```

```
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">}</span>
</li>
</ul>
</li>
<li><span class="Noterminales">A (3)</span>
<ul>
<li><span class="terminales">eof</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</body>
</html>
```

Caso de Prueba # 3

Código fuente

```
/*
    CASO # 3: Probando todas las sentencias
*/

let z1                          int  *= 10  ;
let f_11                        boolean = false ;
let _cad                        string  ;
```

```
let n2                          int     = 5;
let l2                          boolean = true;
input z1;

/* Comentario de prueba */

if(z1&& l2)_cad="HELLO WORLD";
n2 *= z1 + 378;

print                           44
                                  *
                                  z1
                                  *
                                  n2;

/* Funcion sin valor de retorno */
function funcionMyFun (boolean var2)
{
                                l2 =  var2;
                                if (l2) z1 = funcionMyFun (var2);
                                varglobal = 1099;
                                return;
}

if (f_11 == false)
    print varglobal;

switch (z1) {
                                default:
                                funcionMyFun(l2);
                                print "probando switch sin case";
}
```

Parse

```
Descendente
1    5     41    55    15    18    21    24    29    25    22    19    16    1     5
     42    54    15    18    21
24   32    25    22    19    16    1     5     43    53    1     5     41    54
     15    18    21    24    29    25
22   19    16    1     5     42    54    15    18    21    24    31    25    22
     19    16    1     6     10    1
4    15    18    21    24    27    33    25    22    19    17    18    21    24
     27    33    25    22    19    16
8    12    15    18    21    24    30    25    22    19    16    1     6     8
```

```
     13    21    24    27    33    25
23   24    29    25    22    1     6     9     15    18    21    24    29    26
     27    33    26    27    33    25
22   19    16    2     44    46    47    42    50    51    6     8     12    15
     18    21    24    27    33    25
22   19    16    51    4     15    18    21    24    27    33    25    22    19
     16    8     12    15    18    21
24   27    34    35    15    18    21    24    27    33    25    22    19    16
     38    25    22    19    16    51
6    8     12    15    18    21    24    29    25    22    19    16    51    6
     11    40    52    1     4     15
18   21    24    27    33    25    22    20    21    24    32    25    22    19
     16    9     15    18    21    24
27   33    25    22    19    16    1     7     21    24    27    33    25    22
     57    58    6     8     14    35
15   18    21    24    27    33    25    22    19    16    38    60    6     9
     15    18    21    24    30    25
22   19    16    62    3
```

Árbol

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" 'http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd'>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1"/>
<title>Arbol</title>
<link rel="stylesheet" href="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/jquery.treeview.css" />
<link rel="stylesheet" href="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/screen.css" />
<script src="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/jquery.js" type="text/javascript"></script>
<script src="http://www-lt.ls.fi.upm.es/procesadores/Software/VASt/jquery.treeview.js" type="text/javascript"></script>
<link rel="stylesheet" type="text/css" href="CSS.css">
<style type="text/css">
.filetree span.Noterminales { color: Brown;}
.filetree span.terminales { color: Red;}
</style>
<script>
$(document).ready(function(){
$("#browser").treeview();
});
</script>
</head>
<body>
<h4>Árbol resultado de:</h4>
<h4>Gramática: E:\Grettell\Universidad\UPM\Segundo Curso\Primer Semestre\Procesadores de Lenguaje\Práctica\Proyecto\Visualizador Arboles Sintacticos\GramaticaAS.txt</h4>
<h4>Parse: E:\Grettell\Universidad\UPM\Segundo Curso\Primer Semestre\Procesadores de Lenguaje\Práctica\Proyecto\Visualizador Arboles Sintacticos\Casos de Prueba E2\Caso 3\parseC3.txt</h4>
<ul id="browser" class="filetree">

<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (5)</span>
```

```
<ul>
<li><span class="terminales">let</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">T (41)</span>
<ul>
<li><span class="terminales">int</span>
</li>
</ul>
</li>
<li><span class="Noterminales">N (55)</span>
<ul>
<li><span class="terminales">>*=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (29)</span>
<ul>
<li><span class="terminales">cteEntera</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
```

```
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (5)</span>
<ul>
<li><span class="terminales">let</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">T (42)</span>
<ul>
<li><span class="terminales">boolean</span>
</li>
</ul>
</li>
<li><span class="Noterminales">N (54)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (32)</span>
<ul>
<li><span class="terminales">false</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
```

```
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (5)</span>
<ul>
<li><span class="terminales">let</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">T (43)</span>
<ul>
<li><span class="terminales">string</span>
</li>
</ul>
</li>
<li><span class="Noterminales">N (53)</span>
<ul>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (5)</span>
<ul>
<li><span class="terminales">let</span>
</li>
```

```
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">T (41)</span>
<ul>
<li><span class="terminales">int</span>
</li>
</ul>
</li>
<li><span class="Noterminales">N (54)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (29)</span>
<ul>
<li><span class="terminales">cteEntera</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
```

```
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (5)</span>
<ul>
<li><span class="terminales">let</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">T (42)</span>
<ul>
<li><span class="terminales">boolean</span>
</li>
</ul>
</li>
<li><span class="Noterminales">N (54)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (31)</span>
<ul>
<li><span class="terminales">true</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
```

```
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (10)</span>
<ul>
<li><span class="terminales">input</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (4)</span>
<ul>
<li><span class="terminales">if</span>
</li>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
```

```
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (17)</span>
<ul>
<li><span class="terminales">&&</span>
</li>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
```

```
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Sp (12)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
```

```
<ul>
<li><span class="Noterminales">P (30)</span>
<ul>
<li><span class="terminales">cadena</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
```

Sp (13)

*=

U (21)

V (24)

P (27)

id

Pp (33)

lambda

Vp (25)

lambda

Up (23)

+

V (24)

P (29)

cteEntera

Vp (25)

lambda

Up (22)

lambda


```
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (9)</span>
<ul>
<li><span class="terminales">print</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (29)</span>
<ul>
<li><span class="terminales">cteEntera</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (26)</span>
<ul>
<li><span class="terminales">*</span>
</li>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (26)</span>
<ul>
<li><span class="terminales">*</span>
</li>
<li><span class="Noterminales">P (27)</span>
<ul>
```

id

Pp (33)

lambda

Vp (25)

lambda

Up (22)

lambda

Rp (19)

lambda

Ep (16)

lambda

;

A (2)

F (44)

```
<ul>
<li><span class="terminales">function</span>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">H (46)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">D (47)</span>
<ul>
<li><span class="Noterminales">T (42)</span>
<ul>
<li><span class="terminales">boolean</span>
</li>
</ul>
</li>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">K (50)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="terminales">{</span>
</li>
<li><span class="Noterminales">C (51)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Sp (12)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
```

P (27)

id

Pp (33)

lambda

Vp (25)

lambda

Up (22)

lambda

Rp (19)

lambda

Ep (16)

lambda

;

C (51)

B (4)

```
<ul>
<li><span class="terminales">if</span>
</li>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
```

```
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Sp (12)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (34)</span>
<ul>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">L (35)</span>
<ul>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
```

```
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Q (38)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
```

```
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">C (51)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Sp (12)</span>
<ul>
<li><span class="terminales">=</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (29)</span>
<ul>
<li><span class="terminales">cteEntera</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
```

```
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">C (51)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (11)</span>
<ul>
<li><span class="terminales">return</span>
</li>
<li><span class="Noterminales">X (40)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
```

```
</ul>
</li>
<li><span class="Noterminales">C (52)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">}</span>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (4)</span>
<ul>
<li><span class="terminales">if</span>
</li>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
```

```
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (20)</span>
<ul>
<li><span class="terminales">==</span>
</li>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (32)</span>
<ul>
<li><span class="terminales">false</span>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
```

```
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="Noterminales">S (9)</span>
<ul>
<li><span class="terminales">print</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
```

```
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">A (1)</span>
<ul>
<li><span class="Noterminales">B (7)</span>
<ul>
<li><span class="terminales">switch</span>
</li>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
<li><span class="terminales">{</span>
</li>
```

```
<li><span class="Noterminales">Z (57)</span>
<ul>
<li><span class="terminales">default</span>
</li>
<li><span class="terminales">:</span>
</li>
<li><span class="Noterminales">O (58)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (8)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Sp (14)</span>
<ul>
<li><span class="terminales">(</span>
</li>
<li><span class="Noterminales">L (35)</span>
<ul>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (27)</span>
<ul>
<li><span class="terminales">id</span>
</li>
<li><span class="Noterminales">Pp (33)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
```

```
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Q (38)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">)</span>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Op (60)</span>
<ul>
<li><span class="Noterminales">B (6)</span>
<ul>
<li><span class="Noterminales">S (9)</span>
<ul>
<li><span class="terminales">print</span>
</li>
<li><span class="Noterminales">E (15)</span>
<ul>
<li><span class="Noterminales">R (18)</span>
<ul>
<li><span class="Noterminales">U (21)</span>
<ul>
<li><span class="Noterminales">V (24)</span>
<ul>
<li><span class="Noterminales">P (30)</span>
<ul>
<li><span class="terminales">cadena</span>
</li>
```

```
</ul>
</li>
<li><span class="Noterminales">Vp (25)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Up (22)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Rp (19)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Ep (16)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">;</span>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="Noterminales">Op (62)</span>
<ul>
<li><span class="terminales">lambda</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
<li><span class="terminales">}</span>
</li>
</ul>
```

```
</li>
<li><span class="Noterminales">A (3)</span>
<ul>
<li><span class="terminales">eof</span>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</li>
</ul>
</body>
</html>
```

Caso de Prueba # 3

Código fuente

```
/*
    CASO DE PRUEBA #1
*/

print "Introduce el resultado de 6+6*2: ";
let a_1 int = 0;

input a_1;

let num int = 5;

if (a_1 == 18)
    print "Bien!!";
```

```
if (a_1 == 18)    print "Introduce un número para sumarle a 18: ";
if (a_1 == 18)    input num;


switch (a_1) {

    case 24:
        print ":( es 6*2 = 12 y luego 12 + 6 = 18 no 24";
        print "Introduce un número para sumarle a 24: ";
        input num;
        break;
    default:
        print "Vaya " + a_1 + " no es correcto.";
}


function Suma int (int a, int b) {

    j= a + b;
    return j;
    /* La función finaliza y devuelve el valor entero de la expresión */
}

print a_1 + " + " + num + " = ";
print  Suma(a_1,num);
```

INCORRECTOS

Caso de Prueba # 4

Código fuente

```
/*
    CASO DE PRUEBA #4: Estructura de switch incorrecta
*/


print "Introduce el resultado de 6+6*2: ";
let a_1 int = 0;

input a_1;

let num int = 5;

function funcPrint string () {
    print "Bien!!";
```

```
    print "Introduce un número para sumarle a 18: ";
    input num;
}

if (a_1 == 18) funcPrint();

switch (a_1) {

    case num:
        print ":( es 6*2 = 12 y luego 12 + 6 = 18 no 24";
        print "Introduce un número para sumarle a 24: ";
        input num;
        break;
    default:
        print "Vaya " + a_1 + " no es correcto.";
}


function Suma int (int a, int b) {

    j= a + b;
    return j;
    /* La función finaliza y devuelve el valor entero de la expresión */
}

print a_1 + " + " + num + " = ";
print  Suma(a_1,num);
```

Error

```
Línea 23 - Código de error 100:
        ERROR SINTÁCTICO - Token id no esperado. Se esperaba cteEntera
```

Caso de Prueba # 5

Código fuente

```
/*
    CASO DE PRUEBA #5: Estructura de funcion incorrecta
*/
```

```
print "Introduce el resultado de 6+6*2: ";
let a_1 int = 0;

input a_1;

let num int = 5;



function funcPrint string (let) {
    print "Bien!!";
    print "Introduce un número para sumarle a 18: ";
    input num;
}

if (a_1 == 18) funcPrint();

switch (a_1) {

    case 24:
        print ":( es 6*2 = 12 y luego 12 + 6 = 18 no 24";
        print "Introduce un número para sumarle a 24: ";
        input num;
        break;
    default:
        print "Vaya " + a_1 + " no es correcto.";
}



function Suma int (int a, int b) {

    j= a + b;
    return j;
    /* La función finaliza y devuelve el valor entero de la expresión */
}

print a_1 + " + " + num + " = ";
print  Suma(a_1,num);
```

Error

```
Línea 14 - Código de error 121:
        ERROR SINTÁCTICO - Token let no es una declaración válida de un argumento
de una función
```

Caso de Prueba # 6

Código fuente

```
/*
    CASO DE PRUEBA #6: Declarar una función como sentencia de un if
*/


print "Introduce el resultado de 6+6*2: ";
let a_1 int = 0;

input a_1;

let num int = 5;


if (true) function funcPrint string () {
    print "Bien!!";
    print "Introduce un número para sumarle a 18: ";
    input num;
}

if (a_1 == 18) funcPrint();

switch (a_1) {

    case 24:
        print ":( es 6*2 = 12 y luego 12 + 6 = 18 no 24";
        print "Introduce un número para sumarle a 24: ";
        input num;
        break;
    default:
        print "Vaya " + a_1 + " no es correcto.";
}


function Suma int (int a, int b) {

    j= a + b;
    return j;
    /* La función finaliza y devuelve el valor entero de la expresión */
}

print a_1 + " + " + num + " = ";
```

```
print  Suma(a_1,num);
```

Error

```
Línea 14 - Código de error 103:
        ERROR SINTÁCTICO - Token function no pertenece al primer elemento de una sentencia simple válida
```