

<>

Definición

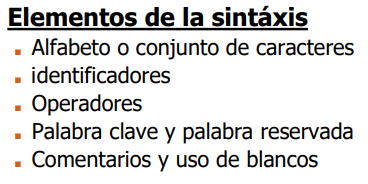
Opcional

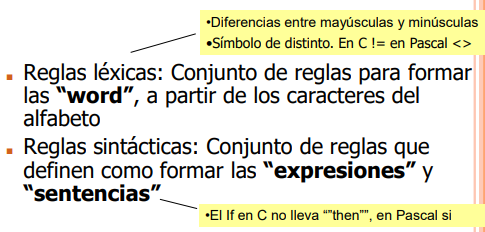
\*

[ ] 

Selección

Palabra no terminal

2) La sintaxis es muy importante porque nos proporcionan los mecanismos para determinar el significado y la validez de un programa, tanto a los programadores como a la computadora.

3)

4)

Palabra clave o keywords, son palabras claves que tienen un significado dentro de un contexto.

Palabra reservada, son palabras claves que además no pueden ser usadas por el programador como identificador de otra entidad.

¿A qué son equivalentes en la definición de una gramática?

En cuanto a la definición de una gramática equivale a las palabras terminales o no terminales.

En cuanto al BNF, las “palabras” reservadas podrían ser [], {}, | , ::=, etc.

Ejemplos en:

Ada: abort declare goto out select

C: auto break case char const continue default

Ruby: else elsif end false ensure

Python: if else elif from lambda return

5)

¿A que se refiere con componente?

Es ambigua porque tomando una expresión, por ejemplo “12”, podemos tomar varios “caminos” para “resolverla”.

1er camino: 1 es <digito> y 2 es <numero\_entero>

2do camino: 1 es <numero\_entero> y 2 es <digito>

6)

G= ( N, T, S, P)

N = { <palabra>, <letra> }

T = {a, b, c, d, etc}

S = <palabra>

P = {

<palabra> ::= <letra> | <letra><palabra>

<letra> ::= a..z | A..Z

}

7)

G= ( N, T, S, P)

N = { <real>,<entero>, <decimal>, <digito>}

T = {0,1,2,3,4,5,6,7,8,9}

S = <real>

P = {

<real> ::= <entero> [‘.’ <decimal>]

<entero> := [(+|-)] <digito> {<digito>}\*

<decimal> := <digito> {<digito>}\*

}

BNF

G= ( N, T, S, P)

N = { <real>,<entero>, <decimal>, <digito> , <numero>}

T = {0,1,2,3,4,5,6,7,8,9}

S = <real>

P = {

<real> ::= <entero> | <entero> “,” <decimal>

<entero> ::= “+” <numero> | “-“ <numero> | <numero>

<numero> ::= <digito> | <digito> <numero>

}

}

8)

BNF

G = ( N, T, S, P)

N = { <palabra>, <letra>, <oracion> }

S = { <oracion> }

T = { a..z, A..Z, “ ” }

P = {

<oracion> ::= <palabra> | <palabra> “ ” <oracion>

<palabra> ::= <letra>|<letra><palabra>

<letra> := a..Z

}

“Conceptos de lenguajes”

<oracion> ::= <palabra> “ ” <oracion>

::= <letra><palabra>“ “<palabra>“ ”<oracion>

C<letra><palabra>” ”<letra><palabra>“ ”<palabra>

Co<letra><palabra>” ”D<letra>” ”<letra><palabra>

Con<letra><palabra>“ ”De” ”l<letra><palabra>

Conc<letra><palabra>“ ”De” ”le<letra><palabra>

Conce<letra><palabra>“ ”De” ”len<letra><palabra>

Concep<letra><palabra>“ ”De” ”leng<letra><palabra>

Concept<letra><palabra>“ ”De” ”lengu<letra><palabra>

Concepto<letra>“ ”De” ”lengua<letra><palabra>

Conceptos “ ”De” ”lenguaj<letra><palabra>

Conceptos “ ”De” ”lenguaje<letra>

Conceptos “ ”De” ”lenguajes

Diagrama

<oracion> = palabra

<palabra> = letra

letra

<letra> = a..Z

/////////

Diagrama

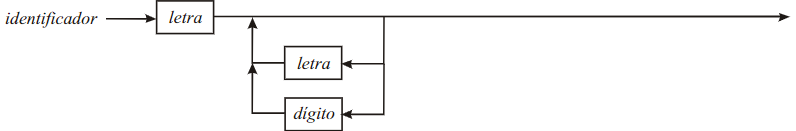
palabra

<oracion> =

<letra> = a..Z

///////////////////

9)

Identificador genérico:

Identificadores de Python:

G= ( N, T, S, P)

N = { <variable>,<letras>,<caracteres>}

T = {ascii}

S = <variable>

P = {

<variable> ::= <letras><caracteres> | <letras>

<letras> ::= “A..Z” | “a..z” | “\_”

<caracteres> ::= todos los ascii

}

10)

G= ( N, T, S, P)

N = { <expresion>, <terminos>, <valor>, <variable>, <letra>, <string>, <numero>, <digito>, <caracter>}

T = {a..z, A..Z, 0..9}

S = <expresion>

P = {

<expresion> ::= <termino> [(+|-) <expresion>]

<termino> ::= <valor> [(\*|/) <termino>]

<valor> ::= <variable> | <numero>

<variable> ::= <letra> {<string>}\*

<letra> ::= a..z |A..Z

<string> ::= {<numero>}+| {<carácter>}+

<numero> ::= {<digito>}+

<digito> ::= 0..9

<carácter> ::= ascii }

11)

(i= IN 1..10) debería ser “(“ <variable> “in” <digito>|<variable>”..”<digito|variable> “)”

La definición de <bloque> parece ambigua

12)

G= ( N, T, S, P)

N = {<etiqueta\_div>, <atributo>, <nombre\_atributo>, <cadena>, <bloque>}

T = { ascii y los atributos (alta paja escribirlos todos)}

S = <etiqueta\_div>

P = {

<etiqueta\_div> ::= ‘<div’ {‘ ‘ <atributo>}\* ‘>’ <bloque> ‘</div>’

<atributo> := <nombre\_atributo> ‘ =” ’ [<cadena>] ‘ ” ’

<nombre\_atributo> := ‘title’ | ‘id’ | ‘class’ | etc

<cadena> := {ascii}+

<bloque> := jajxDDDDD }

14) función para javascript

G= ( N, T, S, P)

N = {}

T = {}

S = <cabecera>

P = {

<cabecera> ::= “function ” <variable> “(“ <parametro> {“,” <parametro>}\* “)” “{“ <sentencias> “}”

<variable> ::= <letra> {<caracter>}\*

<letra> ::= a..z | A..Z | “$” | “\_”

<caracter> ::= ascii

<parametro> ::= <variable> | <numero>

<numero> ::= [(+|-)] {<digito>}+ [“,”{<digito>}+]

<digito> ::= 0..9

<sentencias> ::= jajaXD

}