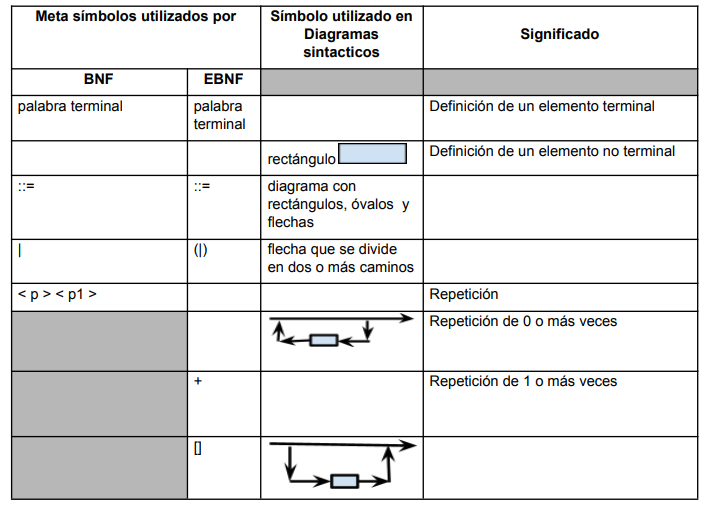
Practica 2   
CPLP



1. La definición de la sintaxis y la semántica de un lenguaje de programación proporcionan mecanismos para que una persona o una computadora pueda decir: ■ Si el programa es válido y ■ Si lo es, qué significa

Elementos de la sintaxis:

* Alfabeto o conjunto de caracteres
* Identificadores
* Operadores
* Comentarios y uso de blancos
* Palabra clave y palabra reservada

1. Reglas léxicas: Conjunto de reglas para formar las “word”, a partir de los caracteres del alfabeto.

Reglas sintácticas: Conjunto de reglas que definen cómo formar a partir de esas palabras, las “expresiones” y “sentencias”.

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

Ventajas de su uso: ■ Permiten al compilador y al programador expresarse claramente ■ Hacen los programas más legibles y permiten una rápida traducción

Son Equivalentes al conjunto de símbolos terminales de la gramatica

Un ejemplo de palabra reservada podria ser WHILE en Python.

1. Los componentes son:

* N: El conjunto de los símbolos no termina. En este caso son:
  + Número entero y Dígito
* T: El conjunto de los símbolos terminales. En este caso son:
  + {0,1,2,3,4,5,,6,7,8,9}
* S: El Símbolo distinguido de la gramática de N:
  + Numero\_Entero
* P: El conjunto de producciones:
  + <numero\_entero> ::= <digito> <numero\_entero> | <numero\_entero> <digito> | <digito>
  + <digito> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Es ambigua ya que para numero entero, define dos veces lo mismo una vez por izquierda y otra por derecha.

Deberia ser: <numero\_entero> ::= <digito> | <digito> <numero\_entero>

1. N = { <palabra>, <letra> }

T = {a-z, A-Z}

S = <palabra>

P = {<palabra> ::= <letra>, <palabra> | <letra>

<letra> ::= {a|b|c|d|e|f|g|h|i|j|k|l|m|n|ñ|o|p|q|r|s|t|v|w|x|y|z|

A|B|C|D|E|F|G|H|I|J|K|L|M|N|Ñ|O|P|Q|R|S|T|V|W|X|Y|Z}

}

1. EBNF:

N = {<numero\_real>, <digito>}

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

S = <numero\_real>

P = { <numero\_real> ::= [-] | <digito> {<digito>} \* | [. | <digito> {<digito>} \*]

<digito> ::= 0|1|2|3|4|5|6|7|8|9 }

BNF:

N = {<numero\_real>, <digito>, <num>}

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

S = <numero\_real}

P = {<numero\_real> ::= <num> | “-” <num> | <num> “,” <num> | “-” <num> “,” <num>|

num ::= <digito> <num> | <digito>

<digito> ::= 0|1|2|3|4|5|6|7|8|9

}

Las diferencias radican en la utilización de

[ ] elemento optativo puede o no estar

(|) selección de una alternativa

{} repetición

\* 0 o más veces + 1 o más veces

1. cuaderno
2. cuaderno

N = {<expresion>, <operador>, <operando>, <digito>, <identificador>,<letra>}

T= {0,1,2,3,4,5,6,7,8,9,-.+./.\*,a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,A,B,C,D,E,F,G,H,I,J,K,L,M.,N,O,P,Q,R,S,T,U,V,W,X,Y,Z}

S = <expresion>

P = { <expresion> ::= { <identificador>, {<operador>, <identificador>}\*

<operando> ::= {<digito> | <digito>\*}

<operador> ::= {-|+|\*|/}

<identificador> ::={ <letra> | <letra> {<digito>}\*}

<letra> :== {a|b|c|d|e|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z|A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|,Q|R|S|T|U|V|W|X|Y|Z}

<dígito> ::= {0|1|2|3|4|5|6|7|8|9|a|b|c|d|e|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z|A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|,Q|R|S|T|U|V|W|X|Y|Z}

}



N = {<expresion\_numerica>, <operador>, <operando>, <digito>, <identificador>,<letra>, <operador\_con\_prioridad>, <operador\_sin\_prioridad> }

T = {0,1,2,3,4,5,6,7,8,9,-.+./.\*,a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,A,B,C,D,E,F,G,H,I,J,K,L,M.,N,O,P,Q,R,S,T,U,V,W,X,Y,Z}

S = <expresion\_numerica>

P= {

<operador\_con\_prioridad> ::= {x|/}

<operador\_sin\_prioridad> ::= {+,-}

<expresion\_numerica> ::= <expresion> [<operador\_sin\_prioridad>, <expresion>]\*

<expresion> ::= <identificador> [<opeador\_con\_prioridad> <identificador>]\*

<identificador> ::= <letra> [<letra> | <digito>]\*

<letra> ::= a|b|c|d|e|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z|A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|,Q|R|S|T|U|V|W|X|Y|Z

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

<operando> ::= <numero> [<numero>]\*

<numero> ::= 0|1|2|3|4|5|6|7|8|9

}



N = {<expresion\_numerica>, <operador>, <operando>, <digito>, <identificador>,<letra>, <operador\_con\_prioridad>, <operador\_sin\_prioridad>, <termino>, <expresion\_entre\_parentesis>, <expresion>}

T = {0,1,2,3,4,5,6,7,8,9,-.+./.\*,a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,A,B,C,D,E,F,G,H,I,J,K,L,M.,N,O,P,Q,R,S,T,U,V,W,X,Y,Z}

S = <expresion\_numerica>

P= {

<expresion\_entre\_parentesis> ::= <expresion> [<operador\_sin\_prioridad> <expresion>]\*

<expresion\_numerica> ::= <terminos> [<operador\_sin\_prioridad> <términos>]\*

<terminos> ::= <expresión> | <expresion\_entre\_parentesis>

<operador\_con\_prioridad> ::= {x|/}

<operador\_sin\_prioridad> ::= {+,-}

<expresión> ::= <identificador> [<operador\_con\_prioridad> <identificador>]\*

<identificador> ::= <letra> [<letra> | <digito>]\*

<letra> ::= a|b|c|d|e|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z|A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|,Q|R|S|T|U|V|W|X|Y|Z

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

<operando> ::= <numero> [<numero>]\*

<numero> ::= 0|1|2|3|4|5|6|7|8|9

}

1. Cuando define P, define un bucle For y toma la sentencia entera de For como un No terminal, cuando este no lo es.

No esta definido T ni tampoco S

Cuando define <cadena>, pone que una opcion es <otro>, el cual nunca esta definido  
En <bloque> define recursividad por izquierda y derecha lo cual hace que la gramática sea ambigua

En <sentencia> hace referencia a símbolos no terminales que nunca fueron definidos Y otros que no estan definidos enel conjunto de símbolos no terminales (N)

1. G = {N,T, S, P}

N = {<div>, <atributo>, <elemento>, <valor>}

T = {a,b,c,d,e,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,Q,R,S,T,U,V,W,X,Y,Z,0,1,2,3,4,5,6,7,8,9}

S = <div>

P = {

<div> ::= < div [<atributo> = <valor>] >

<elemento> ::= {img | p | a | input}

<atributo> ::={ class | id | style | title | lang}

<valor> ::= <letra> | <letra> {<digito>}\*

<letra> ::= a|b|c|d|e|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z|A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|,Q|R|S|T|U|V|W|X|Y|Z

<digito> ::= a|b|c|d|e|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z|A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|,Q|R|S|T|U|V|W|X|Y|Z|0|1|2|3|4|5|6|7|8|9

1. Esto no puede hacerse ya que no hay una regla semantica que permita filtrar los numeros primos. Si se quisiese realizar esto, podriamos aplicar dos soluciones:

**Limitar la gramática a una lista finita de primos**: Podríamos enumerar explícitamente los primeros números primos hasta cierto punto.

**Agregar una restricción semántica**: Especificar que los números generados por <numero> deben cumplir la definición de primalidad (ser mayores que 1 y no tener divisores distintos de 1 y sí mismos).



G = {NTSP}

N = { <funcion>, <cuerpo>, <estructura>, <nombre> }

T = { a,b,c,d,e,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,Q,R,S,T,U,V,W,X,Y,Z,0,1,2,3,4,5,6,7,8,9}

S = <funcion>

P = {

<funcion> ::= “def” <nombre> “(“ {[<nombre>] [,] }\* ”)” <cuerpo> [“return” { <nombre> [,] }\*]

<nombre> ::= <letra> | <letra> {<digito>}\*

<cuerpo> ::= [<estructura>]\*

<estructura> ::= “no la define porque tienen la misma complejidad que el problema que se esta definiendo”

}