SINTACTICO DESCENDENTE LL(1):

Implementación de la gramática S -> ( S ) | n

void S (void) {

switch (token) {

case (:

*match* ( ( );

*S;*

*emparejar* ( ) )

*break,*

case n:

*match* (n);

*break*;

}

}

Un analizador descendente recursivo y predictivo sólo se puede usar con gramáticas LL(1)

Un análisis sintáctico descendente comienza con el símbolo inicial en la pila y la entrada llena y concluye la pila y la entrada vacías. Se pueden realizar 2 acciones:

* **hacer concordar** un toquen en la parte superior de la pila con el siguiente de entrada
* **reemplazar** un no terminal de la parte superior de la pila usando una regla

Una gramática es LL1 si:

* No es ambigua: Es decir, no existe ninguna cadena a la que se le pueda asociar dos árboles de análisis sintáctico diferentes.. Ej ambigua y recursiva: **E-> E+E | num**
* No es recursiva a izquierdas: (A-> Aa|b). Si lo es, puede intentar eliminarse:
  + Eliminación de recursión por la izquierda:
    - Dada **A->Aa|b** obtenemos:
    - **A->bA’  
      A’->aA’|ε**
  + Factorización por la izquierda:
    - Dada **A->ab| ac** obtenemos
    - **A-> aA’**   
       **A’ -> b|c**
* La intersección de sus conjuntos predicción es vacía.

**Conjunto Primero (X):**

Si X es un terminal o **ε,** Primero(X) = {X}  
Si X -> X1 …Xn Primero(X) contiene:

* Primero(X1) – **ε**
* Si todos los conjuntos Primero (X1)… Primero(Xi) contienen **ε**, Primero(X) contiene Primero(Xi+1)-{ **ε** }
* Si todos los Xi contienen **ε** , Primero(X) contiene **ε**

**Conjunto siguiente:**

Se hacen pasadas hasta que no haya más cambios en los conjuntos:

* Si A es el símbolo inicial, Siguiente(A) contiene $
* Si hay una producción B->x A y, Siguiente(A) contiene Primero(y) – **ε**
* Si hay una producción B-> x A y tal que **ε** está en Primero(y) (o A es el último símbolo de la parte derecha), entonces Siguiente(A) contiene Siguiente(B)

**Conjunto predicción:**

* Si **ε**Prim(), Pred(<A> Prim()
* Si **ε**Prim(), Pred(<A> Prim() – { **ε**} } Sig(<A>)

**SINTACTICO DESCENDENTE SLR(1):**

Las gramáticas se aumentan con un símbolo inicial S’ -> S

Un análisis sintáctico ascendente comienza con la pila vacía y la entrada llena y concluye con el símbolo ampliado en la pila y la entrada vacía. Se pueden realizar 2 acciones: **desplazar** el siguiente token de la entrada a la parte superior de la pila o **reducir** una cadena de terminales y no terminales de la parte superior de la pila.