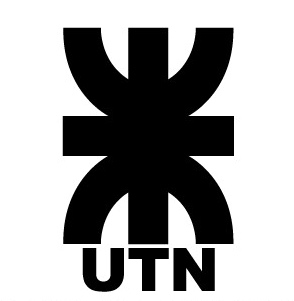
**UTN-FRD**

**Trabajo Práctico Nro. 2**

**Sintaxis y semántica de los lenguajes**

**Alumno:** Julian Vazquez

**Docente:** Hernán Miranda.

**Corrección de la gramática:**

SD(ListaSentencias → ListaSentencias;Sentencia) = {“si”,“repetir”,“id”,“leer”,“mostrar”,“func”}

SD(ListaSentencias → Sentencia) = {“si”,“repetir”,“id”,“leer”,“mostrar”,“func”}

cómo   
SD(ListaSentencias → ListaSentencias;Sentencia)∩SD(ListaSentencias → Sentencia) ≠ ∅  
Luego la gramática no es LL1.

Probamos eliminar recursividad izquierda:  
ListaSentencias → ListaSentencias;Sentencia | Sentencia  
ListaPar → ListaPar “;” “id” | “id”  
Expresion2 → Expresion2 “opsuma” Termino | Termino  
Termino → Termino “opmult” Factor | Factor  
  
Estas producciones resultan luego:

ListaSentencias → Sentencia ListaSentencias\_prime  
ListaSentencias\_prime → ; Sentencia ListaSentencias\_prime / 𝝀

ListaPar → “id” ListaPar\_Prime  
ListaPar\_Prime → “;” “id” ListaPar\_Prime / 𝝀

Expresion2 →Termino Expresion2\_Prime  
Expresion2\_Prime →”opsuma” Termino Expresion2\_Prime / 𝝀

Termino → Factor Termino\_Prime  
Termino\_Prime → “opmult” Factor Termino\_Prime / 𝝀

Al hacer esto sigue sin ser LL1 puesto que:  
SD(SentenciaSi → “si” Expresion “entonces” ListaSentencias “sino” ListaSentencias “finsi”)∩SD(SentenciaSi → “si” Expresion “entonces” ListaSentencias “finsi”) ≠ ∅

Probamos Factorizar las siguientes producciones:

SentenciaSi → “si” Expresion “entonces” ListaSentencias “sino” ListaSentencias “finsi” | “si” Expresion “entonces” ListaSentencias “finsi”

Expresion → Expresion2 “oprel” Expresion2 | Expresion2

Estas producciones resultan:

SentenciaSi → “si” Expresion “entonces” ListaSentencias SentenciaSi\_Prime  
SentenciaSi\_Prime → “sino” ListaSentencias “finsi”/“finsi”

Expresion → Expresion2 Expresion\_Prime  
Expresion\_Prime → “oprel” Expresion2 / 𝝀

**La Gramática resultante es:**

VT = {id, num, si, entonces, sino, finsi, repetir, hasta, equal, leer, mostrar, func, finfunc, (, ), ;, oprel, opsuma, opmult}   
VN = {Program, ListaSentencias, Sentencia, SentenciaSi, SentenciaRepetir, SentenciaAsig, SentenciaLeer, SentenciaMostrar, SentenciaFun,Proc, ListaPar, Expression, Expresion2, Factor, Termino,ListaSentencias\_Prime,SentenciaSi\_Prime,ListaPar\_Prime,Expresion\_Prime, Expresion2\_Prime,Termino\_Prime}  
S = Program  
P= {  
Program →ListaSentencias  
ListaSentencias → Sentencia ListaSentenciasPrima  
ListaSentenciasPrima → “;” Sentencia ListaSentenciasPrima | 𝝀  
Sentencia → SentenciaSi | SentenciaRepetir | SentenciaAsig | SentenciaLeer | SentenciaMostrar | SentenicaFun  
SentenciaSi → “si” Expresion “entonces” ListaSentencias SentenciaSiPrima  
SentenciaSiPrima → “sino” ListaSentencias “finsi” | “finsi”  
SentenciaRepetir → “repetir” ListaSentencias “hasta” Expresion  
SentenciaAsig → “id” “equal” Expresion  
SentenciaLeer → “leer” “id”  
SentenciaMostrar → “mostrar” Expresion  
SentenciaFun → “func” Proc “finfunc”  
Proc → “id” “(“ ListaPar “)” ListaSentencias  
ListaPar → “id” ListaParPrima  
ListaParPrima → “;” “id” ListaParPrima | 𝝀  
Expresion → Expresion2 ExpresionPrima  
ExpresionPrima → “oprel” Expresion2 | 𝝀  
Expresion2 →Termino Expresion2Prima  
Expresion2Prima →”opsuma” Termino Expresion2Prima | 𝝀  
Termino → Factor TerminoPrima  
TerminoPrima → “opmult” Factor TerminoPrima | 𝝀  
Factor → “(“ Expresion “)” | “num” | “id”  
}