

Práctica 5 - Recursividad

Objetivo: Revisar la gramática de su propuesta de lenguaje para asegurar la generación de árboles sintácticos por el método descendente.

Instrucciones:

1. Escriba la gramática completa de su lenguaje y valide que pueda generar todas las cadenas que desea analizar en su código.
2. Factorice la gramática y especifique cuáles sentencias se “redujeron” o aclare que no fue necesario demostrando por qué.

Gramática

```

LInst → L Instr Instr | Instr | ε
Instr → Asig | Funct | Cond
Asig → digit id ⇒ (Num) | text id ⇒ (/string/) | bool id ⇒ (BData)
Num → int | dec
BData → Cero | Uno
Funct → sh(Var_Type)
Cond → SI | Comp | ( LInst ) | O ( LInst ) | Para | int | ( LInst )
Comp → Var_Type Oper Var_Type
Var_Type → digit | text | id | BData
Oper → < | > | = | ! =
  
```



3. Elimine la recursividad por la izquierda y escriba la gramática ampliada. Si no presenta recursividad especifíquelo.

Eliminación de Previsividad.

$\downarrow \text{Inst} \rightarrow \text{Instr } \downarrow \text{Instr}$

$\downarrow \text{Inst} \rightarrow \text{Instr } \downarrow \text{Inst} \mid \text{Inst} \mid \epsilon$

$\text{Instr} \rightarrow \text{Asig} \mid \text{Func} \mid \text{Cond}$

$\text{Asig} \rightarrow \text{digit id} \Rightarrow (\text{Num}) \mid \text{text id} \Rightarrow (/string/) \mid \text{bool id} \Rightarrow (\text{BData})$

$\text{Num} \rightarrow \text{int} \mid \text{dec}$

$\text{BData} \rightarrow \text{Cero} \mid \text{Uno}$

$\text{Func} \rightarrow \text{sh}(\text{Var_Type})$

$\text{Cond} \rightarrow \text{si} \mid \text{Comp} \mid (\text{L Inst}) \mid \text{O}(\text{L Inst}) \mid \text{Para} \mid \text{int} \mid (\text{L Inst})$

$\text{Comp} \rightarrow \text{Var_Type Oper Var_Type}$

$\text{Var_Type} \rightarrow \text{digit} \mid \text{text} \mid \text{id} \mid \text{BData}$

$\text{Oper} \rightarrow < \mid > \mid = \mid ! =$

No hay Factorización

4. Asegúrese de que la gramática final no es ambigua eligiendo dos sentencias: una sencilla y una semi compleja (condición, repetición) para su representación. Aunque eso no lo asegure del todo, será suficiente para la práctica.

Ejemplo

digit a == (10)	LI → IU	→ digit id ⇒ (int)	LI'
	LI → IU	→ digit id ⇒ (int)	LI'
	LI → IU	→ digit id ⇒ (int)	func LI'
	LI → IU	→ digit id ⇒ (int)	sh(Var_Type) LI'
	LI → IU	→ digit id ⇒ (int)	sh(id) LI'
sh(a)	LI → IU	→ digit id ⇒ (int)	sh(id) LI'
SI a == 10 C	LI → IU	→ digit id ⇒ (int)	sh(id) Card LI'
	LI → IU	→ digit id ⇒ (int)	sh(id) LI' Comp (int) LI'
sh(C/Hok; mando)	LI → IU	→ digit id ⇒ (int)	sh(id) SI var_type Oper var_type (not) LI'
	LI → IU	→ digit id ⇒ (int)	sh(id) SI id Oper var_type (not) LI'
	LI → IU	→ digit id ⇒ (int)	sh(id) SI a Oper var_type (not) LI'
	LI → IU	→ digit id ⇒ (int)	sh(id) SI a == var_type (not) LI'
	LI → IU	→ digit id ⇒ (int)	sh(id) SI a == digit (not) LI'
	LI → IU	→ digit id ⇒ (int)	sh(id) SI a == 10 (not) LI'
	LI → IU	→ digit id ⇒ (int)	sh(id) SI a == 10 (func) LI'
	LI → IU	→ digit id ⇒ (int)	sh(id) SI a == 10 (sh(var_type)) LI'
	LI → IU	→ digit id ⇒ (int)	sh(id) SI a == 10 (sh(Hok; mando)) E

OK

5. Entregue en este documento los elementos de cada uno de los puntos, el punto inicial requiere demostración, solo especificar su gramática.

