**REGLAS ORIGINALES:**

1. **<Programa>** → PROGRAMA id ; <decl\_var> <instrucciones> .
2. **<decl\_var>** → VAR <lista\_id> : <tipo> ; <decl\_v>
3. **<decl\_var>** → λ
4. **<decl\_v>** → <lista\_id> : <tipo> ; <decl\_v>
5. **<decl\_v>** → λ
6. **<lista\_id>** → id <resto\_listaid>
7. **<resto\_listaid>** → , <lista\_id>
8. **<resto\_listaid>** → λ
9. **<Tipo>** → <tipo\_std>
10. **<Tipo>** → VECTOR [num] DE <Tipo\_std>
11. **<Tipo\_std>** → ENTERO
12. **<Tipo\_std>** → REAL
13. **<Tipo\_std>** → BOOLEANO
14. **<instrucciones>** → INICIO <lista\_inst> FIN
15. **<lista\_inst>** → <instrucción> ; <lista\_inst>
16. **<lista\_inst>** → λ
17. **<instrucción>** → INICIO <lista\_inst> FIN
18. **<instrucción>** → <inst\_simple>
19. **<instrucción>** → <inst\_es>
20. **<instrucción>** →  SI <expresion> ENTONCES <instrucción> SINO <instrucción>
21. **<instrucción>** →  MIENTRAS <expresión> HACER <instrucción>
22. **<inst\_simple>** → id <resto\_instsimple>
23. **<resto\_instsimple>** → opasigna <expresión>
24. **<resto\_instsimple>** → [ <expr\_simple> ] opasigna <expresión>
25. **<resto\_instsimple>** → λ
26. **<variable>** → id <resto\_var>
27. **<resto\_var>** → [ <expr\_simple> ]
28. **<resto\_var>** → λ
29. **<inst\_es>** → LEE ( id )
30. **<inst\_es>** → ESCRIBE ( <expr\_simple>)
31. **<expresión>** → <expr\_simple> <expresiónPrime>
32. **<expresiónPrime>** → oprel <expr\_simple>
33. **<expresiónPrime>** → λ
34. **<expr\_simple>** → <término> <resto\_exsimple>
35. **<expr\_simple>** → <signo> <término> <resto\_exsimple>
36. **<resto\_exsimple>** → opsuma <término> <resto\_exsimple>
37. **<resto\_exsimple>** → O <término> <resto\_exsimple>
38. **<resto\_exsimple>** → λ
39. **<término>** → <factor> <resto\_term>
40. **<resto\_term>** → opmult <factor> <resto\_term>
41. **<resto\_term>** → Y <factor> <resto\_term>
42. **<resto\_term>** → λ
43. **<factor>** → <variable>
44. **<factor>** → num
45. **<factor>** → ( <expresión> )
46. **<factor>** → NO <factor>
47. **<factor>** → CIERTO
48. **<factor>** → FALSO
49. **<signo>** → +
50. **<signo>** → -

**ATRIBUTOS PARA TIPO DE VARIABLE**

1. **<Programa>** → PROGRAMA id *{si equivaleAReservada(id.lexema) o tablasimbolos.contiene(id.lexema) entonces error  
   sino   
    tablaSimbolos.inserta(id.lexema, “programa”)  
    tablaSimbolos.insertainfo(id.lexema, “clase”, “programa”)  
    fin si}*; <decl\_var> <instrucciones> .
2. **<decl\_var>** → VAR <lista\_id> : <tipo> ; {  
   *para v en <lista\_id>.lista hacer  
    si equivaleAReservada(v) o tablaSimbolos.contiene(v) entonces error  
    sino   
    tablaSimbolos.inserta(v, <tipo>.t)  
    tablaSimbolos.insertainfo(v, “clase”, <tipo>.clase)  
    si <tipo>.clase == “vector” entonces tablaSimbolos.insertainfo(v, “longitud”, <tipo>.longitud)  
    fin si   
   fin para*} <decl\_v>
3. **<decl\_var>** → λ
4. **<decl\_v>** → <lista\_id> : <tipo> ; {  
   *para v en <lista\_id>.lista hacer  
    si equivaleAReservada(v) o tablaSimbolos.contiene(v) entonces error  
    sino   
    tablaSimbolos.inserta(v, <tipo>.t)  
    tablaSimbolos.insertainfo(v, “clase”, <tipo>.clase)  
    si <tipo>.clase == “vector” entonces tablaSimbolos.insertainfo(v, “longitud”, <tipo>.longitud)  
    fin si   
   fin para*} <decl\_v>
5. **<decl\_v>** → λ
6. **<lista\_id>** → id *{<resto\_listaid>.lh = [id.lexema]}* <resto\_listaid> *{<lista\_id>.lista = <resto\_listaid>.lista}*
7. **<resto\_listaid>** → , <lista\_id> *{<resto\_listaid>.lista = <resto\_listaid>.lh ++ <lista\_id>.lista}*
8. **<resto\_listaid>** → λ *{<resto\_listaid>.lista = <resto\_listaid>.lh}*
9. **<Tipo>** → <tipo\_std> *{<Tipo>.t = <tipo\_std>.t; <Tipo>.clase = “variable”, <Tipo>.longitud = 0}*
10. **<Tipo>** → VECTOR [num] DE <Tipo\_std> *{<Tipo>.t = <tipo\_std>.t; <Tipo>.clase = “vector”; <Tipo>.longitud = num.valor}*
11. **<Tipo\_std>** → ENTERO *{<tipo\_std>.t = “entero”}*
12. **<Tipo\_std>** → REAL *{<tipo\_std>.t = “real”}*
13. **<Tipo\_std>** → BOOLEANO *{<tipo\_std>.t = “booleano”}*