## Gramática Atribuida – Comprobación de Tipos

|  |  |  |
| --- | --- | --- |
| **Nodo** | **Predicados** | **Reglas Semánticas** |
| programa → definicion:definicion\* |  |  |
| retorno → tipo:tipo | tipo ≠ IdentType |  |
| cuerpo → defvariable:defVariable\* sentencia:sentencia\* |  |  |
| defCampo → nombre:String tipo:tipo |  |  |
|  |  |  |
| defVariable:definicion → nombre:String tipo:tipo ambito:int |  |  |
| defFuncion:definicion → nombre:String parametros:defVariable\* retorno:retorno cuerpo:cuerpo |  |  |
| defEstructura:definicion → nombre:String defcampo:defCampo\* |  |  |
|  |  |  |
| intType:tipo → λ |  |  |
| realType:tipo → λ |  |  |
| charType:tipo → λ |  |  |
| identType:tipo → nombre:String |  |  |
| arrayType:tipo → dimension:literalInt tipo:tipo |  |  |
|  |  |  |
| while:sentencia → condicion:expresion cierto:sentencia\* | condición.tipo == IntType |  |
| ifElse:sentencia → condicion:expresion cierto:sentencia\* falso:sentencia\* | condición.tipo == IntType |  |
| return:sentencia → expresion:expresion | expresión == null ⇔ definición.retorno == null  expresión ≠ null ⇔ definición.retorno ≠ null  expresión.tipo == definición.retorno.tipo |  |
| print:sentencia → expresion:expresion | expresión ≠ null  expresión.tipo ≠ IdentType |  |
| println:sentencia → expresion:expresion | expresión.tipo ≠ IdentType |  |
| printsp:sentencia → expresion:expresion | expresión ≠ null  expresión.tipo ≠ IdentType |  |
| read:sentencia → expresion:expresion | expresión.modificable == TRUE  expresión.tipo ≠ IdentType |  |
| asigna:sentencia → left:expresion right:expresion | left.tipo ≠ IdentType  left.modificable == TRUE  left.tipo == right.tipo |  |
| llamadaFuncionSentencia:sentencia → nombre:String argumentos:expresion\* | llamadaFuncionSentencia.definicion.parametros.size == argumentos.size  llamadaFuncionSentencia.definicion  .parametrosi.tipo == argumentosj.tipo  i==j |  |
|  |  |  |
| expresionAritmetica:expresion → left:expresion operador:String right:expresion | left.tipo == right.tipo | expresionAritmetica.modificable = FALSE  expresionAritmetica.tipo = left.tipo = right.tipo |
| expresionBooleana:expresion → left:expresion operador:String right:expresion | left.tipo == right.tipo  operador ∈ (&&, ||) ⇒ left.tipo = IntType | expresionBooleana.modificable = FALSE  expresionBooleana.tipo = IntType |
| invocacion:expresion → nombre:String argumentos:expresion\* | invocacion.definicion.parametros.size == argumentos.size  invocacion.definicion.parametrosi.tipo == argumentosj.tipo  i==j | invocación.tipo = invocacion.definicion.retorno.tipo  invocacion.modificable = FALSE |
| variable:expresion → nombre:String |  | variable.tipo = variable.definicion.tipo  variable.modificable = TRUE |
| literalInt:expresion → valor:String |  | literalInt.tipo = IntType  literalInt.modificable = FALSE |
| literalReal:expresion → valor:String |  | literalReal.tipo = RealType  literalReal.modificable = FALSE |
| literalChar:expresion → valor:String |  | literalChar.tipo = CharType  literalChar.modificable = FALSE |
| varArray:expresion → identificacion:expresion posicion:expresion | posición.tipo == IntType  identificación.tipo == ArrayType | varArray.tipo = identificación.tipo.tipo  varArray.modificable = TRUE |
| cast:expresion → tipo:tipo expresion:expresion | tipo ≠ IdentType  expresion.tipo ≠ IdentType  tipo ≠ expresion.tipo | cast.tipo = tipo  cast.modificable = FALSE |
| navega:expresion → expresion:expresion nombre:String | expresion.tipo == IdentType  estructuras[expresion.nombre].campos[nombre] ≠null | navega.tipo = estructuras[expresion.nombre].campos[nombre].tipo  navega.modificable = TRUE |
| conParentesis:expresion → expresion:expresion |  | conParentesis.tipo = expresion.tipo  conParentesis.modificable = expresion.modificable |
| negacion:expresion → expresion:expresion | expresion.tipo == IntType | negacion.modificable = FALSE  negacion.tipo = IntType |
| menosUnario:expresion → expresion:expresion |  | menosUnario.tipo = expresion.tipo  menosUnario.modificable = expresion.modificable |
|  |  |  |

Recordatorio de operadores (para cortar y pegar): ⇒ ⇔ ≠ ∅ ∈ ∉ ∪ ∩ ⊂ ⊄ ∑ ∃ ∀

### Atributos

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Categoría Sintáctica** | **Nombre del atributo** | **Tipo Java** | **Heredado/Sintetizado** | **Descripción** |
| expresion | tipo | Tipo | Sintetizado |  |
| expresion | modificable | boolean | Sintetizado |  |
|  |  |  |  |  |