**Attribute Grammar**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nodo** | **Predicados** | **Reglas Semánticas** |  |
| **program** →*definitions*:definition\* |  |  |  |
|  |  |  |  |
| **varDefinition**:definition→*name*:String |  |  |  |
| *type*:type |  |  |  |
| **structDefinition**:definition→ |  |  |  |
| *name*:varType *definitions*:structField\* |  |  |  |
| **funDefinition**:definition→*name*:String |  |  |  |
| *params*:definition\* *return\_t*:type | params.type ∈ tiposSimples |  |  |
| *definitions*:varDefinition\* | return\_t ∈ tiposSimples |  |  |
| *sentences*:sentence\* |  |  |  |
| **structField**:definition→*name*:String |  |  |  |
| *type*:type |  |  |  |
|  |  |  |  |
| **intType**:type→ λ |  |  |  |
| **realType**:type→ λ |  |  |  |
| **charType**:type→ λ |  |  |  |
| **varType**:type→*type*:String |  |  |  |
| **voidType**:type→ λ |  |  |  |
| **arrayType**:type→*size*:intConstant | size > 0 |  |  |
| *type*:type |  |  |
|  |  |  |
| **structType**:type→*fields*:structField\* |  |  |  |
| **errorType**:type→ λ |  |  |  |
|  |  |  |  |
| **print**:sentence→*expression*:expression | expression.type ∈ tiposSimples |  |  |
| **printsp**:sentence→ | expression.type ∈ tiposSimples |  |  |
| *expression*:expression |  |  |
|  |  |  |
| **println**:sentence→*expression*:expression | expression.type ∈ tiposSimples |  |  |
| **read**:sentence→*expression*:expression | expression.type ∈ tiposSimples |  |  |
| expression.modificable |  |  |
|  |  |  |
| **assignment**:sentence→*left*:expression | mismoTipo(left.type, right.type) |  |  |
| left.modificable |  |  |
| *right*:expression | left.type ∈ tiposSimples |  |  |
|  |  |  |
|  | expression.type ∈ tiposSimples |  |  |
| **return**:sentence→*expression*:expression | mismoTipo(expression.type, | return.definition.hasReturn = true |  |
|  | return.definition.return\_t) |  |  |
| **ifElse**:sentence→*expression*:expression | expression.type == intType |  |  |
| *if\_s*:sentence\* *else\_s*:sentence\* |  |  |  |
| **while**:sentence→*expression*:expression | expression.type == intType |  |  |
| *sentence*:sentence\* |  |  |  |
| **funcInvocation**:sentence→*name*:String | params.type ∈ tiposSimples |  |  |
| *params*:expression\* | params.size == |  |  |
| funcInvocation.definition.params.size |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  | para cada param |  |  |
|  |  |  |
|  | param.type == |  |  |
|  | funcInvocation.definition.param.type |  |  |
|  |  |  |  |
| **variable**:expression→*name*:String |  | variable.type = variable.definicion.typ |  |
|  | variable.modificable = true |  |
|  |  |  |
| **intConstant**:expression→*value*:String |  | intConstant.type = intType |  |
|  | intConstant.modificable = false |  |
|  |  |  |
| **realConstant**:expression→*value*:String |  | realConstant.type = realType |  |
|  | realConstant.modificable = false |  |
|  |  |  |
| **charConstant**:expression→*value*:String |  | charConstant.type = charType |  |
|  | charConstant.modificable = false |  |
|  |  |  |
| **voidConstant**:expression→ λ |  | voidConstatn.type = voidType |  |
|  | voidConstant.modificable = false |  |
|  |  |  |
|  | params.type ∈ tiposSimples |  |  |
|  | params.size == |  |  |
| **funcInvocationExpression**:expression→ | funcInvocation.definition.params.size | funcInvocationExpression.type = |  |
| *name*:String *params*:expression\* | para cada param | funcInvocationExpression.definicion.r |  |
|  | param.type == |  |  |
|  | funcInvocation.definition.param.type |  |  |
| **arithmeticExpression**:expression→ | mismoTipo(left.type, rigth.type) | arithmeticExpression.type = left.type |  |
| *left*:expression *operator*:String |  |
| left.type ∈ {intType, realType} | arithmeticExpression.modificable = fa |  |
| *right*:expression |  |
|  |  |  |
| **logicalExpression**:expression→ | mismoTipo(left.type, rigth.type) | logicalExpression.type = intType |  |
| *left*:expression *operator*:String |  |
| left.type == intType | logicalExpression.modificable = false |  |
| *right*:expression |  |
|  |  |  |
| **unaryExpression**:expression→ | expr.type == intType | unaryExpression.type = intType |  |
| *operator*:String *expr*:expression | unaryExpression.modificable = false |  |
|  |  |
| **comparableExpression**:expression→ | mismoTipo(left.type, rigth.type) | comparableExpression.type = intType |  |
| *left*:expression *operator*:String |  |
| left.type ∈ {intType, realType} | comparableExpression.modificable = f |  |
| *right*:expression |  |
|  |  |  |
| **castExpression**:expression→*type*:type | expr.type ¡= type | castExpression.type = type |  |
| expr.type ∈ tiposSimples |  |
| *expr*:expression | type ∈ tiposSimples | castExpression.modificable = false |  |
|  |  |  |
| **fieldAccessExpression**:expression→ | expr.type == structType | fieldAccessExpression.type = |  |
| (expression.type).field(name) |  |
| *expr*:expression *name*:String |  |
|  | fieldAccessExpression.modificable = t |  |
|  |  |  |
| **indexExpression**:expression→ | expr.type == arrayType | indexExpression.type = expr.type.type |  |
| *expr*:expression *index*:expression | index.type == intType | indexExpression.modificable = true |  |
|  |  |  |  |

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

**Atributos**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nodo/Categoría** | **Nombre** |  | **Tipo** | **Heredado/Sintetiza** | **Descripció** |  |
| **del** |  |
| **Sintáctica** |  | **Java** | **do** | **n** |  |
| **Atributo** |  |  |
|  |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| funcInvocation | definition | FuncDefiniti | Sintetizado |  |  |
| on |  |  |  |
| funcInvocationExpressi | definition | FuncDefiniti | Sintetizado |  |  |
| on | on |  |  |  |
| variable | definition | VarDefinitio | Sintetizado |  |  |
| n |  |  |  |
| expression | type | Type | Sintetizado |  |  |
| expression | modificabl | boolean | Sintetizado |  |  |
|  | e |  |  |  |  |