Attribute Grammar – UO285176

Nodo	Predicados	Reglas Semánticas
program → *definicion*:definicion*		
parametros → *nombre*:String *tipo*:tipo		
variableStruct → *nombre*:String *tipo*:tipo		
defVariable:definicion → *tipo*:tipo *nombre*:String		
defStruct:definicion → *nombre*:String *definicion*:definicion*		
defFuncion:definicion → *nombre*:String *parametros*:parametros* *tipo*:tipo *defvariable*:defVariable* *sentencia*:sentencia*	parametros ∈ tiposSimples tipo ∈ tiposSimples or tipo == null	sentencia.funcion = defFuncion
intTipo:tipo → λ		
realTipo:tipo → λ		
charTipo:tipo → λ		
arrayTipo:tipo → *posicion*:String *tipo*:tipo		
structTipo:tipo → *nombre*:String		
asignacion:sentencia → *izquierda*:expresion *derecha*:expresion	left.tipo == right.tipo left.tipo ∈ tiposSimples left.modificable == true	
print:sentencia → *print*:expresion *printTipo*:String	print.tipo ∈ tiposSimples	
read:sentencia → *read*:expresion	read.tipo ∈ tiposSimples read.modificable = true	
funcionLlamada:sentencia → *nombre*:String *expresion*:expresion*	|funcionLlamada.definicion.expresion| == |expresion|[2] $\forall_i$ expresión$_i$.tipo == funcionLlamada.definicion$_i$.expresion.tipo	
if:sentencia → *condicion*:expresion *if_true*:sentencia* *if_false*:sentencia*	condicion.tipo == intTipo	if_true.funcion == defFuncion if_false.funcion == defFuncion
while:sentencia → *condicion*:expresion *sentencia*:sentencia*	condicion.tipo == intTipo	sentencia.funcion == defFuncion
return:sentencia → *retorno*:expresion	IF return.funcion.tipo == null return.tipo == null ELSE	

	return.tipo == return.funcion.tipo	
expresionAritmetica:expresion → *izquierda*:expresion *operador*:String *derecha*:expresion	left.tipo ∈ tiposSimples left.tipo == right.tipo	expresionAritmetica.tipo = left.tipo expresionAritmetica.modificable = false
expresionLogica:expresion → *izquierda*:expresion *operador*:String *derecha*:expresion	left.tipo == intTipo or left.tipo == realTipo left.tipo == right.tipo	expresionLogica = left.tipo expresionLogica.modificable = false
expresionDistinto:expresion → *not*:expresion	not.tipo == intTipo	expresionDistinto.tipo = intTipo expresionDistinto.modificable = false
variable:expresion → *nombre*:String		variable.tipo = variable.definicion.tipo variable.modificable = true
ident:expresion → *valor*:String		ident.tipo = ident.definicion.tipo variable.modificable = true
litEnt:expresion → *valor*:String		litEnt.tipo = intTipo litEnt.modificable = false
litReal:expresion → *valor*:String		litReal.tipo = realTipo litReal.modificable = false
litChar:expresion → *valor*:String		litChar.tipo = charTipo litChar.modificable = false
cast:expresion → *tipo*:tipo *valor*:expresion	tipo ∈ tiposSimples valor.tipo ∈ tiposSimples tipo ≠ valor.tipo	cast.modificable = false
array:expresion → *nombre*:expresion *valor*:expresion	nombre.tipo == arrayTipo valor.tipo == intTipo	array.tipo = nombre.tipo array.modificable = true
struct:expresion → *nombre*:expresion *campos*:String	nombre.tipo == structTipo	struct.modificable = true
[1]**expresionLlamada**:expresion → *nombre*:String *expresion*:expresion*	expresion.tipo == expresionLlamada.definicion.parametros.tipo expresionLlamada.definicion.tipo ≠ null	expresionLlamada.tipo = expresionLlamada.definicion.tipo

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

Atributos

Nodo/Categoría Sintáctica	Nombre del Atributo	Tipo Java	Heredado/Sintetizado	Descripción
expresion	modificable	boolean	Sintetizado	Indica si la expresión puede aparecer a la izquierda de una asignación
expresion	tipo	Tipo	Sintetizado	Tipo de la expresión (operaciones que admite)
sentencia	Función	DefFuncion	Heredado	Indica la función a la que pertenece la sentencia

Array de tipos auxiliar

Nota: Este array se usar como estructura auxiliar para hacer más ameno comprobar que un tipo sea simple.

tiposSimples = {intTipo, realTipo, charTipo}

1. expresionLlamada representa a la llamada de una función de la forma:

   funcion(a, b);
   sumaVarios(5, 7, nueve);

2. Las || sirven para comprobar el número de parámetros de la llamada es el mismo que el de la definición