2020

Attribute Grammar

DISEÑO DE LENGUAJES DE PROGRAMACIÓN

BORJA RODRIGUEZ LORENZO – UO258643

Nodo	Predicados	Reglas Semánticas
program → ***definitions***:definition*		
defVariable:definition → ***name***:String ***type***:type	Si defVariable.scope == PARAM type ∈ tipoSimple	
defStruct:definition → ***name***:varType ***param***:structField*		
structField:definition → ***name***:String ***type***:type	type ∈ tipoSimple	
defFunc:definition → ***name***:String ***args***:defVariable* ***returnType***:type ***definitions***:defVariable* ***sentences***:sentence*	Si !mismoTipo(returnType, voidType) !defFunc.hasReturn returnType ∈ tipoRetornable	
intType:type → **λ**		
realType:type → **λ**		
charType:type → **λ**		
varType:type → ***name***:String		
arrayType:type → ***size***:intConstant ***type***:type		
errorType:type → **λ**		
voidType:type → **λ**		
assignment:sentence → ***left***:expression ***right***:expression	mismoTipo(left.type, right.type) left.type ∈ tiposSimple left.modificable	
ifElse:sentence → ***expression***:expression ***if_sent***:sentence* ***else_sent***:sentence*	mismoTipo(expression.type, intType)	
while:sentence → ***param***:expression ***sentence***:sentence*	mismoTipo(param.type, intType)	
return:sentence → ***expression***:expression	expression.type ∈ tiposReturn mismoTipo(return.defFunc.returnType, expression.type)	Return.definition.hasReturn = true
read:sentence → ***expression***:expression	expression.type ∈ tiposSimple expression.modificable	
print:sentence → ***expression***:expression	expresssion.type ∈ tipoSimple	

println:sentence → ***expression***:expression	expresssion.type ∈ tipoSimple || expression.type == voidType	
printsp:sentence → ***expression***:expression	expresssion.type ∈ tipoSimple	
funcSentence:sentence → ***name***:String ***args***:expression*	funcSentence.args.size == args.size	
intConstant:expression → ***value***:String		intConstant.type = intType intConstant.modificable = false
realConstant:expression → ***value***:String		realConstant.type = intType realConstant.modificable = false
charConstant:expression → ***value***:String		charConstant.type = intType charConstant.modificable = false
variable:expression → ***value***:String		variable.type = intType variable.modificable = true
voidConstant:expression → λ		voidConstant.type = intType voidConstant.modificable = false
arrayCall:expression → ***index***:expression ***expr***:expression	mismoTipo(index.type, intType) mismoTipo(expr.type, arrayType)	arrayCall.type = expresssion.type.type arrayCall.modificable = true
fieldAccess:expression → ***expression***:expression ***name***:String	mismoTipo(fieldAccess.type, varType) expresssion.type.field != 0	fieldAccess.type = expression.type.field fieldAccess.modificable = true
arithmeticExpr:expression → ***left***:expression ***operator***:String ***right***:expression	mismoTipo(left.type, right.type) left.type ∈ tiposSimple	arithmethicExpr.type = left.type arithmethicExpr.modificable = false
comparationExpr:expression → ***left***:expression ***operator***:String ***right***:expression	mismoTipo(left.type, right.type) left.type ∈ tiposSimple	comparationExpr.type = left.type comparationExpr.modificable = false
logicExpr:expression → ***left***:expression ***operator***:String ***right***:expression	mismoTipo(left.type, right.type) left.type == intType	intConstant.type = intType intConstant.modificable = false
negationExpr:expression → ***operator***:String ***expression***:expression	mismoTipo(expression.type, intType)	negationExpr.type = expression.type negationExpr.modificable = false
castExpr:expression → ***type***:type ***expression***:expression	!mismoTipo(type, expression.type) expression.type ∈ tiposSimple	castExpr.type = type castExpr.modificable = false

	type ∈ tiposSimple	

funcExpr:**expression** → ***name***:**String** ***args***:**expression***	args.size == funcExpr.args.size	funcExpr.type = funcExpr.definition.returnType
		funcExpr.modificable = false

voidExpr:**expression** → **λ**		

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

Atributos

Nodo/Categoría Sintáctica	Nombre del Atributo	Tipo Java	Heredado/ Sintetizado	Descripción
Expression	Type	Type	Sintetizado	Sirve para guardar el tipo
Expression	Modificable	Boolean	Sintetizado	Sirve para saber si un campo es modificable o no
DefFunc	HasReturn	boolean	Heredado	Sirve para guardar si una función tiene retorno o no