Especificación de Código

Función de Código	Plantillas de Código
run[[program]]	run[[program → bloque:bloque*]] = #SOURCE {file} CALL main HALT define[[bloque$_i$]]
define[[bloque]]	define [[funcion → nombre:String parametros:parametro* retorno:tipo* locales:definicion_variable_local* sentencias:sentencia*]] = {nombre}: ENTER {∑localesi.tipo.size} ejecuta[[sentenciasi]] si retorno == VOID RET 0, {∑localesi.tipo.size}, {∑parametrosi.tipo.size}
ejecuta[[sentencia]]	ejecuta [[sentencia_asignacion → izquierda:expr derecha:expr]] = #LINE {end.line} address[[izquierda]] value[[derecha]] STORE<izquierda.tipo>
	ejecuta [[sentencia_print → expresiones:expr]] = #LINE {end.line} value[[expresiones]] OUT< expresiones.tipo> si finCadena != “” PUSHB [[finCadena]] OUTB
	ejecuta [[sentencia_read → expresiones:expr]] = #LINE {end.line} value[[expresiones]] IN<expresiones.tipo>

	ejecuta [[sentencia_if → condicion:expr sentencias:sentencia* sino:sentencia*]] =
	{contadorIF = ++contadorGeneralIF}
	#LINE {end.line}
	if{contadorIf}:
	valor[[condicion]]
	jz else{contadorIf}
	ejecuta[[sentenciasi]]
	jmp finIf{contadorIf}
	else{contadorIf}:
	ejecuta[[sinoi]]
	finIf{contadorIf}:

	ejecuta [[sentencia_while → condicion:expr sentencias:sentencia*]] = {contadorWhile = ++contadorGeneralWhile} #LINE {end.line} while{contadorWhile}: valor[[condicion]] jz finWhile{contadorWhile} ejecuta[[sentenciasi]] jmp while{contadorWhile} finWhile{contadorWhile}:
	ejecuta [[sentencia_llamada_funcion → nombre:String parametros:expr*]] = #LINE {end.line} valor[[parametrosi]] CALL {nombre} si sentencia_llamada_funcion.definicion.retorno != tipoVoid

	POP< sentencia_llamada_funcion.definición.retorno>
	ejecuta [[sentencia_return → expresion:expr]] = #LINE {end.line} si expresion ≠ null valor[[expr]] RET {sentencia_return.funcion.retorno.size}, {∑sentencia_return.funcion.localesi.tipo.size}, {∑sentencia_return.funcion.parametrosi.tipo.size}
valor[[expr]]	valor [[expr_int → string:String]] = PUSH {value}
	valor [[expr_real → string:String]] = PUSHF {value}
	valor [[expr_char → string:String]] = PUSHB {value}
	valor [[expr_ident → string:String]] = address [[expr_ident]] LOAD< expr_ident.type>
	valor [[expr_binaria → izquierda:expr operador:operador derecha:expr]] = value[[izquierda]] value[[derecha]] {operador.instruccion}
	valor [[expr_vector → fuera:expr dentro:expr]] = address[[fuera]] value[[dentro]] PUSHA {tipo.size} MUL ADD LOAD{tipo.size}
	valor[[expr_negada → operador:operador derecha:expr]] = value[[derecha]] {operador.instruccion}
	valor [[expr_punto → izquierda:expr derecha:expr]] = address[[izquierda]] PUSH {derecha.tipo.size} ADD LOAD{tipo.size}
	valor [[expr_parentesis → expr:expr]] = value[[expr]]
	valor[[expr_cast → tipo_convertido:tipo expr:expr]] = value[[expr]] {expr.tipo}2{tipo_convertido}
	valor[[expr_llamada_funcion → nombre:String parametros:expr*]] = valor[[parametrosi]] CALL {nombre}
address[[expr]]	address[[variable → name:String]] = PUSHA {variable.definition.address}