Función	Plantillas de Código
run[[program]]	run[[program → definicion:definicion*]] = #SOURCE {file} CALL main HALT define[[definicion]]
f_2[[parametros]]	f_2[[parametros → nombre:String tipo:tipo]] =
f_3[[variableStruct]]	f_3[[variableStruct → nombre:String tipo:tipo]] = push {variableStruct.address}
define[[definicion]]	f_4[[defVariable → tipo:tipo nombre:String ambito:String]] = #GLOBAL {end.line}
	f_4[[defStruct → nombre:String definicion:definicion*]] = #TYPE {end.line}
	f_4[[defFuncion → nombre:String parametros:parametros* tipo:tipo defvariable:defVariable* sentencia:sentencia*]] = {name}: ENTER {defVariable.type.size} ejecuta[[sentencia]] IF tipo == null RET 0, { ∑defVariable.tipo.size},{∑parametros.tipo.size}
f_5[[tipo]]	f_5[[intTipo → λ]] =
	f_5[[realTipo → λ]] =
	f_5[[charTipo → λ]] =
	f_5[[arrayTipo → posicion:String tipo:tipo]] =
	f_5[[structTipo → nombre:String]] =
ejecuta[[sentencia]]	f_6[[asignacion → izquierda:expresion derecha:expresion]] = #LINE {end.line} address[[izquierda]] valor[[derecha]] STORE

	f_6[[print → print:expresion printTipo:String]] = #LINE {end.line} valor[[print]] OUT
	f_6[[read → read:expresion]] = address[[read]] IN STORE
	f_6[[funcionLlamada → nombre:String expresion:expresion*]] = valor[[expression]] CALL {name} IF funcionLlamada.definicion.tipo != null POP
	f_6[[if → condicion:expresion if_true:sentencia* if_false:sentencia*]] = valor[[condicion]] jz else ejecuta[[if_true]] jmp fin_else else ejecuta[[if_false]] fin_else
	f_6[[while → condicion:expresion sentencia:sentencia*]] = while valor[[condicion]] jz fin_while ejecuta[[sentencia]] jmp while fin_while
	f_6[[return → retorno:expresion]] = IF retorno == null RET 0, {∑ funcion.defvariable.tipo.size}, {∑ funcion.parametros.tipo.size} ELSE RET {expresion.tipo.size}, {∑ funcion.defvariable.tipo.size}, {∑ funcion.parametros.tipo.size}
valor[[expresion]]	valor[[expresionAritmetica → izquierda:expresion operador:String derecha:expresion]] = valor[[derecha]] valor[[izquierda]] IF operador == '+' ADD IF operador == '-' SUB IF operador == '*' MUL IF operador == '/' DIV

	valor[[expresionLogica → izquierda:expresion operador:String derecha:expresion]] = valor[[izquierda]] valor[[derecha]] IF operador == '<' LT IF operador == '>' GT IF operador == '<=' LE IF operador == '>=' GE IF operador == '!=' NE IF operador == '==' EQ
	valor[[expresionLogicaAndOr → izquierda:expresion operador:String derecha:expresion]] = valor[[izquierda]] valor[[derecha]] IF operador == '&&' AND IF operador == '||' OR
	valor[[expresionDistinto → not:expresion]] = valor[[not]] NOT
	valor[[variable → nombre:String]] =
	valor[[ident → valor:String]] = address[[ident]] LOAD
	valor[[litEnt → valor:String]] =
	valor[[litReal → valor:String]] =
	valor[[litChar → valor:String]] =
	valor[[cast → tipo:tipo valor:expresion]] = valor[[valor]] IF tipo == intTipo && expresion.tipo == charTipo b2i IF tipo == intTipo && expresion.tipo == realTipo f2i IF tipo == charTipo && expresion.tipo == intTipo i2b IF tipo == realTipo && expresion.tipo == intTipo i2f
	valor[[array → nombre:expresion valor:expresion]] = address[[nombre]] LOAD

	valor[[struct → nombre:expresion campos:String]] = address[[nombre]] LOAD
	valor[[expresionLlamada → nombre:String expresion:expresion*]] = valor[[expresión]] CALL{nombre}
address[[expresion]]	address[[ident → valor:String]] = IF definición.ambito == GLOBAL PUSHA BP PUSH {definición.address} IF definición.ambito == PARAM PUSH BP PUSH {definición.parametros.address} ADD IF definicion.ambito == LOCAL PUSHA {definicion{dir}}
	address[[struct → nombre:expresion campos:String]] = address[[nombre]] PUSH {nombre.tipo.definicion.defVariable[campos].address} ADD
	address[[array → nombre:expresion valor:expresion]] = address[[nombre]] PUSH {nombre.tipo.size} valor[[valor]] MUL ADD