$programa \rightarrow declaraciones\ funciones$	$STS.push(nuevaTS())$ $STS.push(nuevaTS())$ $dir = 0$ $programa.code = funciones.code$
$declaraciones \rightarrow tipo\ lista\_var\ ;\ declaraciones$	$Tipo = tipo.tipo$
$declaraciones$ $\rightarrow tipo\_registro\ lista\_var\ ;\ declaraciones$	$Tipo = tipo\_registro.tipo$
$declaraciones \rightarrow \varepsilon$	$declaraciones.tipo = base$
$tipo\_registro \rightarrow estructura\ inicio\ declaraciones\ fin$	$STS.push(nuevaTS())$ $STT.push(nuevaTT())$ $Sdir.push(dir)$ $dir = 0$ $dir = Sdir.pop()$ $Ts = STS.pop()$ $Tt = STT.pop()$ $ts.tt = Tt$ $T.tipo = STT.getCima().append('struct', tam, Ts)$
$tipo \rightarrow base\ tipo\_arreglo$	$Base = base.base$ $Tipo.tipo = tipo\_arreglo.tipo$
$base \rightarrow ent$	$base.base = ent$
$base \rightarrow real$	$base.base = real$
$base \rightarrow dreal$	$base.base = dreal$
$base \rightarrow car$	$base.base = car$
$base \rightarrow sin$	$base.base = sin$
$tipo\_arreglo \rightarrow (num)\ tipo\_arreglo_1$	$Si\ num.tipo = ent\ Entonces$ $Si\ num.dir > 0\ Entonces$ $tipo\_arreglo.tipo$ $= TT.append("arreglo", num.dir, tipo\_arreglo_1.tipo)$ $Sino$ $Error("El\ valor\ debe\ ser\ positivo")$ $Fin\ Si$ $Sino$ $Error("El\ valor\ debe\ ser\ entero")$ $Fin\ Si$
$tipo\_arreglo \rightarrow \varepsilon$	$tipo\_arreglo = base$
$lista\_var \rightarrow lista\_var_1, id$	$Si\ !Ts.existe(id)\ Entonces$ $STS.getCima().append(id, dir, Tipo, 'var', nulo, -1)$ $Dir \leftarrow dir + STT.getCima().getTam(Tipo)$ $Sino$ $Error("El\ id\ ya\ fue\ declarado\ anteriormente")$ $Fin\ Si$
$lista\_var \rightarrow id$	$Si\ !Ts.existe(id)\ Entonces$ $STS.getCima().append(id, dir, Tipo, 'var', nulo, -1)$ $Dir \leftarrow dir + STT.getCima().getTam(Tipo)$ $Sino$ $Error("El\ id\ ya\ fue\ declarado\ anteriormente")$ $Fin\ Si$
$funciones \rightarrow tipo\ id(argumentos) inicio$ $declaraciones\ sentencias\ fin\ funciones$	$Si\ !STS.getCima().existe(id)\ Entonces$ $STS.push(nuevaTS())$ $Sdir.push(dir)$ $dir = 0$ $lista\_retorno = nuevaLista()$ $Si\ cmpRet(lista\_retorno, T.tipo)\ Entonces$ $L = nuevaEtiqueta()$ $backpatch(S.nextlist, L)$ $F.code = etiqueta(id)\ ||\ S.code\ ||\ etiqueta(L)$ $Sino$ $Error("El\ valor\ no\ corresponde\ al\ tipo\ de\ la\ funcion")$ $Fin\ Si$

	$STS.pop()$ $dir = Sdir.pop()$ $Sino$ $Error("El\ id\ ya\ fue\ declarado")$ $Fin\ Si$
$funciones \rightarrow \varepsilon$	
$argumentos \rightarrow lista\_arg$	$argumentos.lista = lista\_arg.lista$ $argumentos.num = lista\_arg.num$
$argumentos \rightarrow sin$	$argumentos.lista = nulo$ $argumentos.num = 0$
$lista\_arg \rightarrow lista\_arg_1\ , arg$	$lista\_arg.lista = lista\_arg_1.lista$ $lista\_arg.lista.append(arg.Tipo)$ $lista\_arg.num = lista\_arg.num + 1$
$lista\_arg \rightarrow arg$	$lista\_arg.lista = lista\_arg_1.lista$ $lista\_arg.lista.append(arg.Tipo)$ $lista\_arg.num = lista\_arg.num + 1$
$arg \rightarrow tipo\_arg\ id$	$arg.tipo\ =\ tipo\_arg.tipo$
$tipo\_arg \rightarrow base\ param\_arr$	$Base = base.tipo$ $tipo\_arg.tipo = param\_arr.tipo$
$param\_arr \rightarrow (\ )param\_arr_1$	$param\_arr.tipo = STT.append("array", -, param\_arr_1.tipo)$
$param\_arr \rightarrow \varepsilon$	$param\_arr.tipo = Base$
$sentencias \rightarrow sentencias_1\ sentencia$	$L = nuevaEtiqueta(\ )$ $backpatch(sentencias_1.nextlist, L)$ $sentencias.nextlist = sentencia.nextlist$ $sentencias.code$ $= sentencias_1.code \mid\mid etiqueta(L) \mid\mid sentencia.code$
$sentencias \rightarrow sentencia$	$sentencias.nextlist = sentencia.nextlist$ $sentencias.code = sentencia.code$
$sentencia \rightarrow si\ e\_bool\ entonces\ sentencia_1\ \ fin$	$L = nuevaEriqueta(\ )$ $backpatch(e\_bool.truelist, L)$ $sentencia.nextlist$ $= combinar(e\_bool.falselist, sentencia_1.nextlist)$ $sentencia.code = e\_bool.code \mid\mid etiqueta(L) \mid\mid sentencia_1.code$
$sentencia$ $\rightarrow si\ e\_bool\ entonces\ sentencia_1\ sino\ sentencia_2\ fin$	$L_1 = nuevaEtiqueta(\ )$ $L_2 = nuevaEtiqueta(\ )$ $backpatch(e\_bool.truelist, L_1)$ $backpatch(e\_bool.falselist, L_2)$ $sentencia.nextlist$ $= combinar(sentencia_1.nextlist, sentencia_2.nextlist)$ $sentencia.code = e_{bool}.code \mid\mid etiqueta(L_1) \mid\mid sentencia_1.code$ $\mid\mid gen('goto' sentencia_1.nextlist[0]) \mid\mid etiqueta(L_2)$ $\mid\mid sentencia_2.code$
$sentencia \rightarrow mientras\ e\_bool\ hacer\ sentencia_1\ fin$	$L_1 = nuevaEtiqueta(\ )$ $L_2 = nuevaEtiqueta(\ )$ $backpatch(sentencia_1.nextlist, L_1)$ $backpatch(e\_bool.truelist, L_2)$ $sentencia.nextlist = e\_bool.falselist$ $sentencia.code = etiqueta(L_1) \mid\mid e\_bool.code \mid\mid etiqueta(L_2) \mid\mid$ $sentencia_1.code \mid\mid gen('goto' sentencia_1.nextlist[0])$
$sentencia \rightarrow hacer\ sentencia_1\ mientras\ e\_bool;$	$L_1 = nuevaEtiqueta(\ )$ $L_2 = nuevaEtiqueta(\ )$ $backpatch(sentencia_1.nextlist, L_1)$ $backpatch(e\_bool.truelist, L_2)$ $sentencia.nextlist = e\_bool.falselist$

	$sentencia.code = etiqueta(L_2) \mid\mid sentencia_1.code \mid\mid$ $etiqueta(L_1) \mid\mid e\_bool.code \mid\mid gen('goto' sentencia_1.nextlist[0])$
$sentencia$ $\rightarrow segun\ (variable) hacer\ casos\ predeterminado\ fin$	$L_1 = nuevaEtiqueta()$ $L_2 = nuevaEtiqueta()$ $backpatch(sentencia.truelist, L_1)$ $backpatch(sentencia.falselist, L_2)$ $sentencia.nextlist$ $= combinar(casos.nextlist, predeterminado.nextlist)$ $sentencia.code = variable.code \mid\mid etiqueta(L_1) \mid\mid casos.code \mid\mid$ $gen('goto'\ casos.nextlist[0]) \mid\mid etiqueta(L_2) \mid\mid$ $predeterminado.code$
$sentencia \rightarrow variable := expresion;$	$sentencia.nextlist = null$ $Si\ TS.existe(variable)\ Entonces$ $tipo\_variable = TS.getTipo(variable)$ $t = reducir(expresion.dir, expresion.tipo, tipo\_variable)$ $sentencia.codegen = variable(\quad ' \quad =' t)$ $Sino$ $error(La\ variable\ no\ ha\ sido\ declarada)$ $Fin\ Si$
$sentencia \rightarrow escribir\ expresion;$	$sentencia.code = gen("printf"\ expresion.dir)$ $sentencia.nextlist = null$
$sentencia \rightarrow leer\ variable;$	$sentencia.code = gen("scanf"\ variable.dir)$ $sentencia.nextlist = null$
$sentencia \rightarrow devolver;$	$sentencia.code = gen("return")$ $sentencia.nextlist = null$
$sentencia \rightarrow devolver\ expresion;$	$lista\_retorno.append(expresion.tipo)$ $sentencia.code = gen(returnexpresion.dir)$ $sentencia.nextlist = null$
$sentencia \rightarrow terminar;$	$L = nuevaEtiqueta()$ $sentencia.code = gen('goto'\ L)$ $sentencia.nextlist = nuevaLista()$ $sentencia.nextlist.add(L)$
$sentencia \rightarrow inicio\ sentencias\ fin$	$sentencia.nextlist = sentencias.nextlist$
$casos \rightarrow caso\ num: sentencia\ casos_1$	$L_1 = nuevaEtiqueta()$ $L_2 = nuevaEtiqueta()$ $backpatch(num.truelist, L_1)$ $backpatch(num.falselist, L_2)$ $casos.nextlist = combiar(sentencia.nextlist, casos_1.nextlist)$ $casos.code = num.dir \mid\mid etiqueta(L_1) \mid\mid sentencia.code \mid\mid$ $gen('goto'\ sentencia.nextlist[0]) \mid\mid etiqueta(L_2) \mid\mid casos_1\ .code$
$casos \rightarrow caso\ num: sentencia$	$L_1 = nuevaEtiqueta()$ $backpatch(num.truelist, L_1)$ $casos.code = num.dir \mid\mid etiqueta(L_1) \mid\mid sentencia.code \mid\mid$ $gen('goto'\ sentencia.nextlist[0])$
$predeterminado \rightarrow pred: sentencia$	$L_1 = nuevaEtiqueta()$ $backpatch(num.falselist, L_1)$ $predeterminado.code = num.dir \mid\mid etiqueta(L_1) \mid\mid$ $sentencia.code \mid\mid gen('goto'\ sentencia.nextlist[0])$
$predeterminado \rightarrow \varepsilon$	$predeterminado.code = null$
$e\_bool \rightarrow e\_bool_1\ o\ e\_bool_2$	$L = nuevaEtiqueta()$ $backpatch(e\_bool_1.flaselist, L)$ $e\_bool.truelist = combinar(e\_bool_1.truelist, e\_bool_2.truelist)$ $e\_bool.falselist = e\_bool_2.falselist$ $e\_bool.code = e\_bool_1.code \mid\mid etiqueta(L) \mid\mid e\_bool_2.code$

$e\_bool \rightarrow e\_bool_1 \; y \; e\_bool_2$	$L = nuevaEtiqueta()$ $backpatch(e\_bool_1.truelist, L)$ $e\_bool.truelist = e\_bool_2.truelist$ $e\_bool.falselist = combinar(e\_bool_1.falselist, e\_bool_2.falselist)$ $e\_bool.code = e\_bool_1.code \; \| \| \; etiqueta(L) \; \| \| \; e\_bool_2.code$
$e\_bool \rightarrow no \; e\_bool_1$	$e\_bool.truelist = e\_bool_1.falselist$ $e\_bool.falselist = e\_bool_1.truelist$ $e\_bool.code = e\_bool_1.code$
$e\_bool \rightarrow relacional$	$e\_bool.truelist = relacional.truelist$ $e\_bool.falselist = relacional.falselist$
$e\_bool \rightarrow verdadero$	$t_0 = nuevoIndice()$ $e\_bool.truelist = crearLista(t_0)$ $e\_bool.code = gen('goto' \; t_0)$
$e\_bool \rightarrow falso$	$t_0 = nuevoIndice()$ $e\_bool.falselist = crearLista(t_0)$ $e\_bool.code = gen('goto' \; t_0)$
$relacional \rightarrow relacional_1 > relacional_2$	$relacional.dir = nuevaTemporal$ $relacional.tipo = max \; (relacional_1.tipo, relacional_2.tipo)$ $t_1 = ampliar(relacional_1.dir, relacional_1.tipo, relacional.tipo)$ $t_2 = ampliar(relacional_2.dir, relacional_2.tipo, relacional.tipo)$ $relacional.code = gen(relacional.dir = t_1' >' \; t_2)$
$relacional \rightarrow relacional_1 < relacional_2$	$relacional.dir = nuevaTemporal$ $relacional.tipo = max \; (relacional_1.tipo, relacional_2.tipo)$ $t_1 = ampliar(relacional_1.dir, relacional_1.tipo, relacional.tipo)$ $t_2 = ampliar(relacional_2.dir, relacional_2.tipo, relacional.tipo)$ $relacional.code = gen(relacional.dir = t_1' <' \; t_2)$
$relacional \rightarrow relacional_1 <= relacional_2$	$relacional.dir = nuevaTemporal$ $relacional.tipo = max \; (relacional_1.tipo, relacional_2.tipo)$ $t_1 = ampliar(relacional_1.dir, relacional_1.tipo, relacional.tipo)$ $t_2 = ampliar(relacional_2.dir, relacional_2.tipo, relacional.tipo)$ $relacional.code = gen(relacional.dir = t_1' <=' \; t_2)$
$relacional \rightarrow relacional_1 >= relacional_2$	$relacional.dir = nuevaTemporal$ $relacional.tipo = max \; (relacional_1.tipo, relacional_2.tipo)$ $t_1 = ampliar(relacional_1.dir, relacional_1.tipo, relacional.tipo)$ $t_2 = ampliar(relacional_2.dir, relacional_2.tipo, relacional.tipo)$ $relacional.code = gen(relacional.dir = t_1' >=' \; t_2)$
$relacional \rightarrow relacional_1 <> relacional_2$	$relacional.dir = nuevaTemporal$ $relacional.tipo = max \; (relacional_1.tipo, relacional_2.tipo)$ $t_1 = ampliar(relacional_1.dir, relacional_1.tipo, relacional.tipo)$ $t_2 = ampliar(relacional_2.dir, relacional_2.tipo, relacional.tipo)$ $relacional.code = gen(relacional.dir = t_1' <>' \; t_2)$
$relacional \rightarrow relacional_1 = relacional_2$	$relacional.dir = nuevaTemporal$ $relacional.tipo = max \; (relacional_1.tipo, relacional_2.tipo)$ $t_1 = ampliar(relacional_1.dir, relacional_1.tipo, relacional.tipo)$ $t_2 = ampliar(relacional_2.dir, relacional_2.tipo, relacional.tipo)$ $relacional.code = gen(relacional.dir = t_1' =' \; t_2)$
$relacional \rightarrow expresion$	$relacional.dir = expresion.dir$ $relacional.code = expresion.code$
$expresion \rightarrow expresion_1 + expresion_2$	$expresion.dir = nuevaTemporal$ $expresion.tipo = (expresion_1.tipo, expresion_2.tipo)$ $t_1 = ampliar(expresion_1.dir, expresion_1.tipo, expresion.tipo)$ $t_2 = ampliar(expresion_2.dir, expresion_2.tipo, expresion.tipo)$ $expresion.code = gen(expresion.dir' =' \; t_1' + ' t_2)$
$expresion \rightarrow expresion_1 - expresion_2$	$expresion.dir = nuevaTemporal$

	$expresion.tipo = (expresion_1.tipo, expresion_2.tipo)$ $t_1 = ampliar(expresion_1.dir, expresion_1.tipo, expresion.tipo)$ $t_2 = ampliar(expresion_2.dir, expresion_2.tipo, expresion.tipo)$ $expresion.code = gen(expresion.dir' =' t_1' -' t_2)$
$expresion \rightarrow expresion_1 * expresion_2$	$expresion.dir = nuevaTemporal$ $expresion.tipo = (expresion_1.tipo, expresion_2.tipo)$ $t_1 = ampliar(expresion_1.dir, expresion_1.tipo, expresion.tipo)$ $t_2 = ampliar(expresion_2.dir, expresion_2.tipo, expresion.tipo)$ $expresion.code = gen(expresion.dir' =' t_1' *' t_2)$
$expresion \rightarrow expresion_1 / expression_2$	$expresion.dir = nuevaTemporal$ $expresion.tipo = (expresion_1.tipo, expresion_2.tipo)$ $t_1 = ampliar(expresion_1.dir, expresion_1.tipo, expresion.tipo)$ $t_2 = ampliar(expresion_2.dir, expresion_2.tipo, expresion.tipo)$ $expresion.code = gen(expresion.dir' =' t_1'/' t_2)$
$expresion \rightarrow expresion_1 \% expresion_2$	$expresion.dir = nuevaTemporal$ $expresion.tipo = (expresion_1.tipo, expresion_2.tipo)$ $t_1 = ampliar(expresion_1.dir, expresion_1.tipo, expresion.tipo)$ $t_2 = ampliar(expresion_2.dir, expresion_2.tipo, expresion.tipo)$ $expresion.code = gen(expresion.dir' =' t_1'\%' t_2)$
$expresion \rightarrow (expresion_1)$	
$expresion \rightarrow variable$	$Si\ TS.existe(variable)\ Entonces$ $expresion.dir = variable.dir$ $expresion.tipo = TS.getTipo(variable)$ $Sino$ $Error("La\ variable\ no\ ha\ sido\ declarada")$ $Fin\ Si$
$expresion \rightarrow num$	$expresion.tipo = num.tipo$ $expresion.dir = num.val$
$expresion \rightarrow cadena$	$expresion.tipo = cadena.tipo$ $expresion.dir = TablaDeCadenas.add(cadena.val)$
$expresion \rightarrow caracter$	$expresion.tipo = caracter.tipo$ $expresion.dir = TablaDeCadens.add(caracter.val)$
$variable \rightarrow id\ variable\_comp$	$Si\ TS.existe(id)\ Entonces$ $tipo\_id = TS.getTipo()$ $t = reducir(variable\_comp.dir, variable\_comp.tipo, tipo\_id)$ $Sino$ $Error("El\ id\ no\ ha\ sido\ declarado")$ $Fin\ Si$
$variable\_comp \rightarrow dato\_est\_sim$	$variable\_comp.dir = dato\_est\_sim.dir$ $variable.code = dato\_est\_sim.code$
$variable\_comp \rightarrow arreglo$	$variable\_comp.dir = arreglo.dir$ $variable\_comp.base = arreglo.base$ $variable\_comp.tipo = arreglo.tipo$
$variable\_comp \rightarrow (parametros)$	$variable\_comp.lista = parametros.lista$ $variable\_comp.num = parametros.num$
$dato\_est\_sim \rightarrow dato\_est\_sim\ .id$	$Si\ !TS.existe(id)\ Entonces$ $STS.getFondo().append(id, dir, Tipo)$ $dir \leftarrow dir + STT.getFondo().getTam(Tipo)$ $Sino$ $Error("El\ id\ no\ ha\ sido\ declarado")$
$dato\_est\_sim \rightarrow \varepsilon$	
$arreglo \rightarrow (expresion)$	$t = nuevaTemporal()$ $arreglo.dir = nuevaTemporal()$ $arreglo.tipo = array$

	$arreglo.tam = TT.getTam(expresion.tipo)$ $arreglo.base = expresion.base$ $arreglo.code = gen(t' = 'expresion.dir' * ' arreglo.tam)$
$arreglo \rightarrow arreglo_1(expresion)$	$Si\ TT.getNombre(arreglo_1.tipo) = array\ Entonces$ $t = nuevaTemporal()$ $arreglo.dir = nuevaTemporal()$ $arreglo.tipo = TT.getTipoBase(arreglo_1.tipo)$ $arreglo.tam = TT.getTam(arreglo_1.tipo)$ $arreglo.base = arreglo_1.base$ $arreglo.code = gen(t' = 'expresion.dir' * 'arreglo.tam)$ || $gen(arreglo.dir' = ' arreglo_1.dir' + ' t)$ $Sino$ $Error("La\ variable\ asociada\ no\ es\ un\ arreglo")$ $Fin\ Si$
$parametros \rightarrow lista\_param$	$parametros.lista = lista\_param.lista$ $parametros.num = lista\_param.num$
$parametros \rightarrow \varepsilon$	$parametros.lista = null$ $parametros.num = 0$
$lista\_param \rightarrow lista\_param_1, expresion$	$lista\_param.lista = nuevaLista()$ $lista\_param.lista.append(expresion.tipo)$ $lista\_param.num = lista\_param_1.num + 1$
$lista\_param \rightarrow expresion$	$lista\_param.lista = nuevaLista()$ $lista\_param.lista.append(expresion.tipo)$ $lista\_param.num = 1$