## Especificación de Código

|  |  |
| --- | --- |
| **Función de Código** | **Plantillas de Código** |
| run : Programa → Instruccion\* | run [[ Programa  →  definiciones:Definicion\* ]] =  define[[definicionesi]] |
|  |  |
| define : Definicion → Instruccion\* | define[[ DefinicionFuncion  →  nombre:String  parametros:DefinicionVariable\*   retorno:Tipo  locales:DefinicionVariable\*  sentencias:Sentencia\* ]] =  ejecuta[[sentenciasi]] |
|  |  |
| ejecuta : Sentencia → Instruccion\* | // Hacer solo Asignación y Write |
|  | ejecuta [[ Asignacion  →  left:Expresion  right:Expresion ]] =  #LINE {end.line}  dirección[[left]]  valor[[right]]  STORE |
|  | ejecuta [[ Print  →  expr:Expresion ]] =  #LINE {end.line}  valor[[expr]]  OUT |
|  |  |
| valor: Expresion → Instruccion\* | // Hacer todas las expresiones excepto invocación de funciones  valor[[**LiteralInt** -> valor:String]] =  #LINE{end.line} |
|  | PUSH {valor}  valor[[**LiteralReal** -> valor:String]] =  #LINE{end.line}  PUSH {valor}  valor[[**Caracter** -> valor:String]] =  #LINE{end.line}  PUSH{end.line}  valor[[**Variable** -> nombre:String]] =  #LINE{end.line}  PUSHA {variable.definicion.direccion}  LOAD  valor[[**CampoStruct** -> left:expresion right:String]] =  #LINE{end.line}  PUSHA {left[nombre==”right”].dir}  LOAD  valor[[**CallArray** -> variable:expresion acceso:expresión]] =  #LINE{end.line}  dirección[[variable]]  valor[[acceso]]  PUSH {variable.tipo.size}  MUL  ADD  LOAD  valor[[**ExpresionBinaria** -> left:expresión operador:string right:expresión]] =  #LINE{end.line}  valor[[left]]  valor[[right]]  si operador == ‘+’  ADD  si operador == ‘-‘  SUB  si operador == ‘/’  DIV  si operador == ‘\*’  MUL |
| dirección: expresión -> Intruccion\* | dirección[Variable -> nombre:String]] =  PUSHA {variable.definicion.dir} |

NOTA: Lo que está en naranja es temporal para esta clase y habrá que ampliarlo en la siguiente.