## Especificación de Código

|  |  |  |
| --- | --- | --- |
| **Función de Código** | **Plantillas de Código** | |
| run⟦Programa⟧ | run⟦Programa  →  definiciones:Definicion\*⟧ =  #SOURCE {file}  define⟦definicionesi⟧ | |
|  |  | |
| define⟦Definicion⟧ | define⟦DefinicionFuncion  →  nombre:String  parametros:DefinicionVariable\*   retorno:Tipo  locales:DefinicionVariable\*  sentencias:Sentencia\*⟧ =  ejecuta⟦sentenciasi⟧ | |
|  |  | |
| ejecuta⟦Sentencia⟧ | // Por ahora, hacer solo Asignación y Print | |
|  | ejecuta⟦Asignacion  →  left:Expresion  right:Expresion⟧ =  #LINE {end.line}  /\* Seguir \*/ | |
|  | ejecuta⟦Print  →  expr:Expresion⟧ =  #LINE {end.line}  /\* Seguir \*/ | |
|  |  | |
| valor⟦Expresion⟧ | valor[[constanteEntero  →  valor:int ]] =  PUSH {valor} | |
|  | valor[[constanteReal  →  valor:double ]] =  PUSHF {valor} | |
|  | valor[[constanteChar  →  valor:String ]] =  PUSHB {valor} | |
|  | valor[[variable  →  nombre:String ]] =  address[[variable]]  LOAD<variable.tipo> | |
|  | valor[[expresionAritmetica  →  izq:expresion  operador:String  der:expresion ]] =  value[[izq]]  value[[der]]  si operador == “+”  ADD<arithmeticExpression.type>  si operador == “-”  SUB<arithmeticExpression.type>  si operador == “\*”  MUL<arithmeticExpression.type>  si operador == “/”  DIV<arithmeticExpression.type>  si operador == “%”  ????????????? | |
|  | valor[[expresionLogica  →  izq:expresion  operador:String  der:expresion ]] =  value[[izq]]  value[[der]]  si operador == “&&”  AND  si operador == “||”  OR | |
|  | valor[[comparacion  →  izq:expresion  operador:String  der:expresion ]] =  value[[izq]]  value[[der]]  si operador == “<”  LT  si operador == “>”  GT  si operador == “≥”  GT  value[[izq]]  value[[der]]  EQ  OR  si operador == “≤”  LT  value[[izq]]  value[[der]]  EQ  OR | |
|  | valor[[expresionUnaria  →  expresion:expresion  operador:String ]] =  value[[expresion]]  NOT | |
|  | valor[[conversion  →  nuevoTipo:tipo  expresion:expresion ]] =  value[[expresion]]  si expresion.tipo == TipoEntero  si nuevoTipo == TipoFloat  I2F  si nuevoTipo == TipoChar  I2B  si expresion.tipo == TipoReal  si nuevoTipo == TipoEntero  F2I  si nuevoTipo == TipoChar  F2I  I2B  si expresion.tipo == TipoChar  si nuevoTipo == TipoEntero  B2I  si nuevoTipo == TipoFloat  B2I  I2F | |
|  | valor[[invocacionExpresion  →  nombre:String  params:expresion\* ]] =  valor[[paramsi]] CALL {nombre} |
|  | valor[[accesoArray  →  array:expresion  indice:expresion ]] = |
|  | valor[[accesoCampo  →  struct:expresion  campo:expresion ]] = |

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