## Code Specification

|  |  |
| --- | --- |
| **Función de Código** | **Plantillas de Código** |
| run[[**program**]] | run[[**programa**  →  *definitions*:varDefinition\* *sentences*:sentence\* ]] =  #SOURCE {sourceFile}  metadata[[definitionsi]]  execute[[sentencesi]]  HALT |
|  |  |
| metadata[[**varDefinition**]] | metadata[[**varDefinition**  →  *type*:type  *name*:String ]] =  #GLOBAL {name}: {MAPLType(type)} |
|  |  |
| execute[[**sentence**]] | execute[[**print**  →  *expression*:expression ]] =  #LINE {end.line}  value[[expression]]  OUT<expression.type> |
|  | execute[[**asignment**  →  *left*:expression  *right*:expression ]] =  #LINE {end.line}  address[[left]]  value[[right]]  STORE<left.type> |
|  |  |
| value[[**expression**]] | value[[**arithmeticExpression**  →  *left*:expression  *operador*:String *right*:expression ]] =  value[[left]]  value[[right]]  si operador == “+”  ADD<arithmeticExpression.type>  si operador == “-”  SUB<arithmeticExpression.type>  si operador == “\*”  MUL<arithmeticExpression.type>  si operador == “/”  DIV<arithmeticExpression.type> |
|  | value[[**variable**  →  *name*:String ]] =  address[[variable]]  LOAD<variable.type> |
|  | value[[**intConstant**  →  *value*:String ]] =  PUSH {value} |
|  | value[[**realConstant**  →  *value*:String ]] =  PUSHF {value} |
|  |  |
| address[[**expression**]] | address[[**variable**  →  *name*:String ]] =  PUSHA {variable.definition.address} |
|  |  |

**Nota:**

La notación *Instruccion<expresión de tipo>* representa a la versión adecuada de la instrucción para el tipo indicado.

Ejemplos:

LOAD<int> → LOADI

LOAD<real> → LOADF

**Metadata:**

Todas las líneas con metadatos (prefijadas con el símbolo *#*) son opcionales (ver el tutorial de MAPL para más información).

### Funciones Auxiliares

|  |  |
| --- | --- |
| Función Auxiliar | Definición |
| MAPLType(type) | Si type == intType  MAPLType = “int”  Si type == realType  MAPLType = “float” |