**ESTRUCTURAS DE CÓDIGO CON SENTIDO DEL COMPILADOR**

Las estructuras de código con sentido son generadas a través de los tipos de datos, dónde se evalúan las estructuras ya generadas para que tengan un sentido que es aceptado por este compilador, por ende las estructuras con sentido de este compilador que se pueden generar son las que están en la siguiente tabla:

|  |  |  |
| --- | --- | --- |
| **Estructuras de Declaraciones de puerto (DP) proximity** | | |
| PUERTO **proximity** IDENFICADOR FIN\_DE\_LINEA | port\_A1 proximity proximitySensor! | |
| **Estructuras de Declaraciones de puerto (DP) temperature** | | |
| PURTO **temperature** IDENFICADOR FIN\_DE\_LINEA | port\_A2 temperature temperatureSensor! | |
| **Estructuras de Declaraciones de puerto (DP) LED** | | |
| PUERTO **LED** IDENFICADOR FIN\_DE\_LINEA | port\_A3 LED LEDSensor! | |
| **Estructuras de Declaraciones de puerto (DP) LCD** | | |
| PUERTO **LCD** IDENFICADOR FIN\_DE\_LINEA | port\_A4 LCD LCDSensor! | |
| **Estructuras de Declaraciones de puerto (DP) button** | | |
| PUERTO **button** IDENFICADOR FIN\_DE\_LINEA | port\_A5 button buttonSensor! | |
| **Estructuras de Declaraciones de puerto (DP) motor** | | |
| PUERTO **motor** IDENFICADOR FIN\_DE\_LINEA | port\_B6 motor motorSensor! | |
| **Estructuras de Declaraciones con valor (DCV) string** | | |
| DECLARACION **string** IDENTIFICADOR OPERADOR\_ASIGNACION **IDENTIFICADOR\_CADENA** FIN\_DE\_LINEA | var string holaMundo = 'HolaMundo'! | |
| **Estructuras de Declaraciones con valor (DCV) int** | | |
| DECLARACION **int** IDENTIFICADOR OPERADOR\_ASIGNACION **NUMERO\_ENTERO** FIN\_DE\_LINEA | var int numero1 = 234! | |
| **Estructuras de Declaraciones con valor (DCV) decimal** | | |
| DECLARACION **decimal** IDENTIFICADOR OPERADOR\_ASIGNACION **NUMERO\_REAL** FIN\_DE\_LINEA | var decimal numero2 = 34.56! | |
| **Estructuras de Declaraciones con valor (DCV) boolean** | | |
| DECLARACION **boolean** IDENTIFICADOR OPERADOR\_ASIGNACION **CONDICIONAL** FIN\_DE\_LINEA | var boolean valor3 = true! | |
| **Estructuras de Declaraciones sin valor (DSV) string** | | |
| DECLARACION **string** IDENTIFICADOR FIN\_DE\_LINEA | const string valor1 ! | |
| **Estructuras de Declaraciones sin valor (DSV) int** | | |
| DECLARACION **int** IDENTIFICADOR FIN\_DE\_LINEA | const int valor2 ! | |
| **Estructuras de Declaraciones sin valor (DSV) decimal** | | |
| DECLARACION **decimal** IDENTIFICADOR FIN\_DE\_LINEA | const decimal valor3 ! | |
| **Estructuras de Declaraciones sin valor (DSV) boolean** | | |
| DECLARACION **boolean** IDENTIFICADOR FIN\_DE\_LINEA | const boolean valor4 ! | |
| **Estructura de Asignaciones (A) string** | | |
| **IDENTIFICADOR** OPERADOR\_ASIGNACION **IDENTIFICADOR\_CADENA** FIN\_DE\_LINEA | Valor1 = ‘Hola’ ! | |
| **Estructura de Asignaciones (A) int** | | |
| **IDENTIFICADOR** OPERADOR\_ASIGNACION **NUMERO\_ENTERO** FIN\_DE\_LINEA | Valor2 = 3 ! | |
| **Estructura de Asignaciones (A) decimal** | | |
| **IDENTIFICADOR** OPERADOR\_ASIGNACION **NUMERO\_REAL** FIN\_DE\_LINEA | valor3 = 3.3 ! | |
| **Estructura de Asignaciones (A) boolean** | | |
| **IDENTIFICADOR** OPERADOR\_ASIGNACION **CONDICIONAL** FIN\_DE\_LINEA | Valor4 = true ! | |
| **Estructura de funciones de los motores con valor (FMCV) move** | | |
| **move** SIGNO\_AGRUPACION\_3 **IDENTIFICADOR** SIGNO\_PUNTUACION\_1 **NUMERO\_ENTERO** SIGNO\_PUNTUACION\_1 **IDENTIFICADOR** SIGNO\_AGRUPACION\_4 FIN\_DE\_LINEA | move(motorSensor,34,button1)! | |
| **Estructura de funciones de los motores sin valor (FMCV) start** | | |
| **start** SIGNO\_AGRUPACION\_3 **IDENTIFICADOR** SIGNO\_PUNTUACION\_1 **IDENTIFICADOR** SIGNO\_AGRUPACION\_4 FIN\_DE\_LINEA | start(motorSensor,button1)! | |
| **Estructura de funciones de los motores sin valor (FMCV) restart** | | |
| **restart** SIGNO\_AGRUPACION\_3 **IDENTIFICADOR** SIGNO\_PUNTUACION\_1 **IDENTIFICADOR** SIGNO\_AGRUPACION\_4 FIN\_DE\_LINEA | restart(motorSensor,button1)! | |
| **Estructura de Método de impresora a consola (MIC)** | | |
| **print** SIGNO\_AGRUPACION\_3 **console** SIGNO\_PUNTUACION\_1 (**IDENTIFICADOR \_CADENA** | **IDENTIFICADOR**) SIGNO\_AGRUPACION\_4 FIN\_DE\_LINEA | print(console,'El motor1 se esta moviendo')! | |
| **Estructura de Método de impresora a LCD (MIL)** | | |
| **print** SIGNO\_AGRUPACION\_3 **IDENTIFICADOR** SIGNO\_PUNTUACION\_1 (**IDENTIFICADOR \_CADENA** | **IDENTIFICADOR**) SIGNO\_AGRUPACION\_4 FIN\_DE\_LINEA | print(LCDSensor,'El motor1 se esta moviendo')! | |
| **Estructura de Operaciones (O)** | | |
| **operation** **IDENTIFICADOR** OPERADOR\_ASIGNACION (**NUMERO\_ENTERO** | **NUMERO\_REAL** | **IDENTIFICADOR**) OPERADOR\_ARITMETICO (**NUMERO\_ENTERO** | **NUMERO\_REAL** | **IDENTIFICADOR**) FIN\_DE\_LINEA | operation valor2 = a + b! | |
| **Estructura de las funciones sin parámetros (FSP)** | | |
| **function IDENTIFICADOR** SIGNO\_AGRUPACION\_3 SIGNO\_AGRUPACION\_4 SIGNO\_AGRUPACION\_1 (**FMCV** | **FMSV** | **MIC** | **MIL**)\* SIGNO\_AGRUPACION\_2 | function moverMotor(){  move(motorSensor,60)!  print(console,'El motor1 se esta moviendo')!  } | |
| **Estructura de las funciones con parámetros (FCP) int** | | |
| **function int** **IDENTIFICADOR** SIGNO\_AGRUPACION\_3 **int** **IDENTIFICADOR** (SIGNO\_PUNTUACION\_1 **int** **IDENTIFICADOR**)\* SIGNO\_AGRUPACION\_4 SIGNO\_AGRUPACION\_1 (**O**)\* **return** **IDENTIFICADOR** FIN\_DE\_LINEA SIGNO\_AGRUPACION\_2 | function int sumaNumerosInt(int a, int b){  operation valor2 = a + b!  return valor2!  } | |
| **Estructura de las funciones con parámetros (FCP) decimal** | | |
| **function decimal** **IDENTIFICADOR** SIGNO\_AGRUPACION\_3 **decimal** **IDENTIFICADOR** (SIGNO\_PUNTUACION\_1 **decimal** **IDENTIFICADOR**)\* SIGNO\_AGRUPACION\_4 SIGNO\_AGRUPACION\_1 (**O**)\* **return** **IDENTIFICADOR** FIN\_DE\_LINEA SIGNO\_AGRUPACION\_2 | function decimal sumaNumerosInt(decimal a, decimal b){  operation valor2 = a + b!  return valor2!  } | |
| **Estructura de llamadas de funciones con valor (LFCV)** | | |
| **call** **IDENTIFICADOR** SIGNO\_AGRUPACION\_3 (**NUMERO\_ENTERO** | **NUMERO\_REAL**) (SIGNO\_PUNTUACION\_1 (**NUMERO\_ENTERO** | **NUMERO\_REAL**))\* SIGNO\_AGRUPACION\_4 FIN\_DE\_LINEA | call sumaNumerosInt (5,10)! | |
| **Estructura de llamadas de funciones sin valor (LFSV)** | | |
| **call** **IDENTIFICADOR** SIGNO\_AGRUPACION\_3 SIGNO\_AGRUPACION\_4 FIN\_DE\_LINEA | call moverMotor()! | |
| **Estructura de control Begin (B)** | | |
| **begin** SIGNO\_AGRUPACION\_1 (**LFCV** | **LFSV** | **FMCV** | **FMSV**)\* SIGNO\_AGRUPACION\_2 | begin{  call moverMotor()!  } | |
| **Estructura de Método de los sensores (MS)** | | |
| **call** **IDENTIFICADOR** OPERADOR\_ASIGNACION **IDENTIFICADOR** SIGNO\_PUTUACION\_2 **METODO\_SENSOR** SIGNO\_AGRUPACION\_3 SIGNO\_AGRUPACION\_4 FIN\_DE\_LINEA | call Distancia = proximitySensor.distance()! | |
| **Estructura de Método Delay (MD)** | | |
| **delay** SIGNO\_AGRUPACION\_3 (**IDENTIFICADOR** | **NUMERO\_ENTERO**) SIGNO\_AGRUPACION\_4 FIN\_DE\_LINEA | | delay(1000)! |
| **Estructura de** Método Encender (ME) | | |
| **ligther** SIGNO\_AGRUPACION\_3 **IDENTIFICADOR** SIGNO\_PUNTUACION\_1 **IDENTIFICADOR** SIGNO\_AGRUPACION\_4 FIN\_DE\_LINEA | | ligther(led1,boton1)! |
| **Estructura de** Método Apagar (MA) | | |
| **turn\_off** SIGNO\_AGRUPACION\_3 **IDENTIFICADOR** SIGNO\_PUNTUACION\_1 **IDENTIFICADOR** SIGNO\_AGRUPACION\_4 FIN\_DE\_LINEA | | turn\_off(led1,boton1)! |
| **Estructura de control If Relacional (IR)** | | |
| **if** SIGNO\_AGRUPACION\_3 (**IDENTIFICADOR** | **NUMERO\_ENTERO** | **NUMERO\_REAL**) OPERADOR\_RELACIONAL (**IDENTIFICADOR** | **NUMERO\_ENTERO** | **NUMERO\_REAL**) SIGNO\_AGRUPACION\_4 SIGNO\_AGRUPACION\_1 (**MS** | **MIC** | **MIL** | **A** | **DCV** | **DSV** | **FMCV** | **FMSV** | **LFCV** | **LFSV** | **MD** | **ME** | **MA**)\* SIGNO\_AGRUPACION\_2 | if(numero1>valor2){  call moverMotor()!  } | |
| **Estructura de control If Lógico con dos valores (IL2V)** | | |
| **if** SIGNO\_AGRUPACION\_3 (**IDENTIFICADOR** | **CONDICIONAL**) OPERADOR\_LOGICO (**IDENTIFICADOR** | **CONDICIONAL**) SIGNO\_AGRUPACION\_4 SIGNO\_AGRUPACION\_1 (**MS** | **MIC** | **MIL** | **A** | **DCV** | **DSV** | **FMCV** | **FMSV** | **LFCV** | **LFSV** | **MD** | **ME** | **MA**)\* SIGNO\_AGRUPACION\_2 | if(valor3 and valor4){  call moverMotor()!  } | |
| **Estructura de control If Lógico con un valor (IL1V)** | | |
| **if** SIGNO\_AGRUPACION\_3 (**IDENTIFICADOR** | **CONDICIONAL**) SIGNO\_AGRUPACION\_4 SIGNO\_AGRUPACION\_1 (**MS** | **MIC** | **MIL** | **A** | **DCV** | **DSV** | **FMCV** | **FMSV** | **LFCV** | **LFSV** | **MD** | **ME** | **MA**)\* SIGNO\_AGRUPACION\_2 | if(valor3){  call moverMotor()!  } | |
| **Estructura de control Else Relacional (ER)** | | |
| **if** SIGNO\_AGRUPACION\_3 (**IDENTIFICADOR** | **NUMERO\_ENTERO** | **NUMERO\_REAL**) OPERADOR\_RELACIONAL (**IDENTIFICADOR** | **NUMERO\_ENTERO** | **NUMERO\_REAL**) SIGNO\_AGRUPACION\_4 SIGNO\_AGRUPACION\_1 (**MS** | **MIC** | **MIL** | **A** | **DCV** | **DSV** | **FMCV** | **FMSV** | **LFCV** | **LFSV** | **MD** | **ME** | **MA**)\* SIGNO\_AGRUPACION\_2 **else** SIGNO\_AGRUPACION\_1 (**MS** | **MIC** | **MIL** | **A** | **DCV** | **DSV** | **FMCV** | **FMSV** | **LFCV** | **LFSV** | **MD** | **ME** | **MA**)\* SIGNO\_AGRUPACION\_2 | if(numero1>valor2){  call moverMotor()!  }else{    } | |
| **Estructura de control Else Lógico con dos valores (EL2V)** | | |
| **if** SIGNO\_AGRUPACION\_3 (**IDENTIFICADOR** | **CONDICIONAL**) OPERADOR\_LOGICO (**IDENTIFICADOR** | **CONDICIONAL**) SIGNO\_AGRUPACION\_4 SIGNO\_AGRUPACION\_1 (**MS** | **MIC** | **MIL** | **A** | **DCV** | **DSV** | **FMCV** | **FMSV** | **LFCV** | **LFSV** | **MD** | **ME** | **MA**)\* SIGNO\_AGRUPACION\_2 **else** SIGNO\_AGRUPACION\_1 (**MS** | **MIC** | **MIL** | **A** | **DCV** | **DSV** | **FMCV** | **FMSV** | **LFCV** | **LFSV** | **MD** | **ME** | **MA**)\* SIGNO\_AGRUPACION\_2 | if(valor3 and valor4){    }else{  call moverMotor()!  } | |
| **Estructura de control Else Lógico con un valor (EL1V)** | | |
| **if** SIGNO\_AGRUPACION\_3 (**IDENTIFICADOR** | **CONDICIONAL**) SIGNO\_AGRUPACION\_4 SIGNO\_AGRUPACION\_1 (**MS** | **MIC** | **MIL** | **A** | **DCV** | **DSV** | **FMCV** | **FMSV** | **LFCV** | **LFSV** | **MD** | **ME** | **MA**)\* SIGNO\_AGRUPACION\_2 **else** SIGNO\_AGRUPACION\_1 (**MS** | **MIC** | **MIL** | **A** | **DCV** | **DSV** | **FMCV** | **FMSV** | **LFCV** | **LFSV** | **MD** | **ME** | **MA**)\* SIGNO\_AGRUPACION\_2 | if(valor3){  call moverMotor()!  }else{    } | |
| **Estructura de control Loop (L)** | | |
| **loop** SIGNO\_AGRUPACION\_1 (**MS** | **MIC** | **MIL** | **A** | **DCV** | **DSV** | **FMCV** | **FMSV** | **LFCV** | **LFSV** | **IR** | **IL2V** | **IL1V** | **ER** | **EL2V** | **EL1V** | **MD** | **ME** | **MA**)\* SIGNO\_AGRUPACION\_2 | loop{  if(valor3 and valor4){  call moverMotor()!  }else{    }  call Distancia = proximitySensor.distance()!    } | |