**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**) (Identificador) (Fin de línea) | port\_A1 proximity proximitySensor! | |
| **Estructuras de Declaraciones de puerto (DP) temperature** | | |
| (Puerto) (**temperature**) (Identificador) (Fin de línea) | port\_A2 temperature temperatureSensor! | |
| **Estructuras de Declaraciones de puerto (DP) LED** | | |
| (Puerto) (**LED**) (Identificador) (Fin de línea) | port\_A3 LED LEDSensor! | |
| **Estructuras de Declaraciones de puerto (DP) LED\_RGB** | | |
| (Puerto) (**LED\_RGB**) (Identificador) (Fin de línea) | port\_A4 LED\_RGB LED\_RGBSensor! | |
| **Estructuras de Declaraciones de puerto (DP) LCD** | | |
| (Puerto) (**LCD**) (Identificador) (Fin de línea) | port\_A4 LCD LCDSensor! | |
| **Estructuras de Declaraciones de puerto (DP) button** | | |
| (Puerto) (**button**) (Identificador) (Fin de línea) | port\_A5 button buttonSensor! | |
| **Estructuras de Declaraciones de puerto (DP) motor** | | |
| (Puerto) (**motor**) (Identificador) (Fin de línea) | port\_B6 motor motorSensor! | |
| **Estructuras de Declaraciones con valor (DCV) string** | | |
| (Declaración) (**string**) (Identificador) (Operador asignación) (**Cadena**) (Fin de línea) | var string holaMundo = 'HolaMundo'! | |
| **Estructuras de Declaraciones con valor (DCV) int** | | |
| (Declaración) (**int**) (Identificador) (Operador asignación) (**Numero entero**) (Fin de línea) | var int numero1 = 234! | |
| **Estructuras de Declaraciones con valor (DCV) decimal** | | |
| (Declaración) (**decimal**) (Identificador) (Operador asignación) (**Numero real**) (Fin de línea) | var decimal numero2 = 34.56! | |
| **Estructuras de Declaraciones con valor (DCV) boolean** | | |
| (Declaración) (**boolean**) (Identificador) (Operador asignación) (**Condicionales**) (Fin de línea) | var boolean valor3 = true! | |
| **Estructuras de Declaraciones sin valor (DSV) string** | | |
| (Declaración) (string) (Identificador) (Fin de línea) | const string valor1 ! | |
| **Estructuras de Declaraciones sin valor (DSV) int** | | |
| (Declaración) (int) (Identificador) (Fin de línea) | const int valor2 ! | |
| **Estructuras de Declaraciones sin valor (DSV) decimal** | | |
| (Declaración) (decimal) (Identificador) (Fin de línea) | const decimal valor3 ! | |
| **Estructuras de Declaraciones sin valor (DSV) boolean** | | |
| (Declaración) (boolean) (Identificador) (Fin de línea) | const boolean valor4 ! | |
| **Estructura de Asignaciones (A) Depende del tipo de dato que es la variable** | | |
| (**variable**) (Operador asignación) (**VALORES**) (Fin de línea) | valor3 = 3.3 ! | |
| **Estructura de funciones de los motores con valor (FMCV) move** | | |
| (**move**) (Signo agrupación “(” ) (**Variable de puerto asociado al motor**) (Signo puntuación “,” ) (**Numero entero**) (Signo agrupación “)” ) (Fin De línea) | move(motorSensor,34)! | |
| **Estructura de funciones de los motores con valor (FMCV) start** | | |
| (**start**) (Signo agrupación “(” ) (**Variable de puerto asociado al motor**) (Signo agrupación “)” ) (Fin De línea) | start(motorSensor)! | |
| **Estructura de funciones de los motores con valor (FMCV) restart** | | |
| (**restart**) (Signo agrupación “(” ) (**Variable de puerto asociado al motor**) (Signo agrupación “)” ) (Fin De línea) | restart(motorSensor)! | |
| **Estructura de Método de impresora a consola (MIC)** | | |
| (**print**) (Signo agrupación “(”) (**console**) (Signo puntuación “,” ) (**Cadena** | **Nombre de variable con valor cadena**) (Signo agrupación “)” ) (Fin de línea) | print(console,'El motor1 se esta moviendo')! | |
| **Estructura de Método de impresora a LCD (MIL)** | | |
| (**print**) (Signo agrupación “(”) (**Variable que está asociado a un puerto con el sensor LCD**) (Signo puntuación “,” ) (**Cadena** | **Nombre de variable con valor cadena**) (Signo agrupación “)” ) (Fin de línea) | print(LCDSensor,'El motor1 se esta moviendo')! | |
| **Estructura de Operaciones (O)** | | |
| (**operation**) (**Variable con el tipo de dato acorde al tipo de dato de la operación**) (Operador asignación) (**Variable declarada en los parámetros de la función o un valor acorde al tipo de dato de la variable donde se guardara**) (Operador aritmético) (**Variable declarada en los parámetros de la función o un valor acorde al tipo de dato de la variable donde se guardara**) (Fin de línea) | operation valor2 = a + b! | |
| **Estructura de las funciones sin parámetros (FSP)** | | |
| (**function**) (**Nombre de la función**) (Signo agrupación “(” ) (Signo agrupación “)” ) (Signo agrupación “{” ) (**FMCV | FMSV**)\* (**MIC** | **MIL**)\* (Signo agrupación “}” ) | function moverMotor(){  move(motorSensor,60)!  print(console,'El motor1 se esta moviendo')!  } | |
| **Estructura de las funciones con parámetros (FCP)** | | |
| (**function**) (**Tipo dato a retornar**) (**Nombre de la función**) (Signo agrupación “(” ) (Tipo dato) (**Nombre de la variable parámetro**) ((Signos puntuación “,”) (Tipo dato) (**Nombre de la variable parámetro**))\* (Signo agrupación “)” ) (Signo agrupación “{” ) (**O**)\* (**return**) (**Variable con el mismo tipo de dato a retornar de la función**) (Fin de línea) (Signo agrupación “}” ) | function int sumaNumerosInt(int a, int b){  operation valor2 = a + b!  return valor2!  } | |
| **Estructura de llamadas de funciones con valor (LFCV)** | | |
| (**call**) (**Nombre de la función a llamar**) (Signo agrupación “(” ) (**Valor con el mismo tipo de dato del parámetro de la función**) ((Signo puntuación “,”) (**Valor con el mismo tipo de dato del parámetro de la función**))\* (Signo agrupación “)” ) (Fin de línea) | call sumaNumerosInt (5,10)! | |
| **Estructura de llamadas de funciones sin valor (LFSV)** | | |
| (**call**) (**Nombre de la función a llamar**) (Signo agrupación “(” ) (Signo agrupación “)” ) (Fin de línea) | call moverMotor()! | |
| **Estructura de control Begin (B)** | | |
| (**begin**) (Signo agrupación “{” ) ((**LFCV**) | (**LFSV**) | (**FMCV**) | **FMSV**)\* (Signo agrupación “}” ) | begin{  call moverMotor()!  } | |
| **Estructura de Método de los sensores (MS)** | | |
| (**call**) (**Nombre de variable tipo cadena**) (Operador asignación) (**Variable de puerto asociado a algún sensor**) (Signo puntuación “.” ) (Método sensor) (Signo agrupación “(” ) (Signo agrupación “)” ) (Fin de línea) | call Distancia = proximitySensor.distance()! | |
| **Estructura de Método Delay (MD)** | | |
| (**delay**) (Signo agrupación “(” ) ((**Variable de tipo de dato Entero**) | (**Valor de tipo Numero entero**) (Signo agrupación “)” ) (Fin de línea) | | delay(1000)! |
| **Estructura de control If Relacional (IR)** | | |
| (**if**) (Signo agrupación “(” ) ((**Variable de tipo Entero o Decimal**) | (**Valor de tipo Entero o Decimal**)) ((Operador relacional) | (Operador asignación)) ((**Variable de tipo Entero o Decimal**) | (**Valor de tipo Entero o Decimal**)) (Signo agrupación “)” ) (Signo agrupación “{” ) ((**MS**) | (**MIC**) | (**MIL**) | (**A**) | (**DCV**) | (**DSV**) | (**FMCV**) | (**FMSV**) | (**LFCV**) | (**LFSV**) | (**MD**))\* (Signo agrupación “}” ) | if(numero1>valor2){  call moverMotor()!  } | |
| **Estructura de control If Lógico con dos valores (IL2V)** | | |
| (**if**) (Signo agrupación “(” ) ((**Variable de tipo Booleano**) | (**Valor de tipo Booleano**)) (Operador lógico) ((**Variable de tipo Booleano**) | (**Valor de tipo Booleano**)) (Signo agrupación “)” ) (Signo agrupación “{” ) ((**MS**) | (**MIC**) | (**MIL**) | (**A**) | (**DCV**) | (**DSV**) | (**FMCV**) | (**FMSV**) | (**LFCV**) | (**LFSV**) | (**MD**))\* (Signo agrupación “}” ) | if(valor3 and valor4){  call moverMotor()!  } | |
| **Estructura de control If Lógico con un valor (IL1V)** | | |
| (**if**) (Signo agrupación “(” ) ((**Variable de tipo Booleano**) | (**Valor de tipo Booleano**)) (Signo agrupación “)” ) (Signo agrupación “{” ) ((**MS**) | (**MIC**) | (**MIL**) | (**A**) | (**DCV**) | (**DSV**) | (**FMCV**) | (**FMSV**) | (**LFCV**) | (**LFSV**) | (**MD**))\* (Signo agrupación “}” ) | if(valor3){  call moverMotor()!  } | |
| **Estructura de control Else Relacional (ER)** | | |
| (**if**) (Signo agrupación “(” ) ((**Variable de tipo Entero o Decimal**) | (**Valor de tipo Entero o Decimal**)) ((Operador relacional) | (Operador asignación)) ((**Variable de tipo Entero o Decimal**) | (**Valor de tipo Entero o Decimal**)) (Signo agrupación “)” ) (Signo agrupación “{” ) ((**MS**) | (**MIC**) | (**MIL**) | (**A**) | (**DCV**) | (**DSV**) | (**FMCV**) | (**FMSV**) | (**LFCV**) | (**LFSV**) | (**MD**))\* (Signo agrupación “}” ) (**else**) (Signo agrupación “{” ) ((**MS**) | (**MIC**) | (**MIL**) | (**A**) | (**DCV**) | (**DSV**) | (**FMCV**) | (**FMSV**) | (**LFCV**) | (**LFSV**) | (**MD**))\* (Signo agrupación “}” ) | if(numero1>valor2){  call moverMotor()!  }else{    } | |
| **Estructura de control Else Lógico con dos valores (EL2V)** | | |
| (**if**) (Signo agrupación “(” ) ((**Variable de tipo Booleano**) | (**Valor de tipo Booleano**)) (Operador lógico) ((**Variable de tipo Booleano**) | (**Valor de tipo Booleano**)) (Signo agrupación “)” ) (Signo agrupación “{” ) ((**MS**) | (**MIC**) | (**MIL**) | (**A**) | (**DCV**) | (**DSV**) | (**FMCV**) | (**FMSV**) | (**LFCV**) | (**LFSV**) | (**MD**))\* (Signo agrupación “}” ) (**else**) (Signo agrupación “{” ) ((**MS**) | (**MIC**) | (**MIL**) | (**A**) | (**DCV**) | (**DSV**) | (**FMCV**) | (**FMSV**) | (**LFCV**) | (**LFSV**) | (**MD**))\* (Signo agrupación “}” ) | if(valor3 and valor4){    }else{  call moverMotor()!  } | |
| **Estructura de control Else Lógico con un valor (EL1V)** | | |
| (**if**) (Signo agrupación “(” ) ((**Variable de tipo Booleano**) | (**Valor de tipo Booleano**)) (Signo agrupación “)” ) (Signo agrupación “{” ) ((**MS**) | (**MIC**) | (**MIL**) | (**A**) | (**DCV**) | (**DSV**) | (**FMCV**) | (**FMSV**) | (**LFCV**) | (**LFSV**))\* (Signo agrupación “}” ) (**else**) (Signo agrupación “{” ) ((**MS**) | (**MIC**) | (**MIL**) | (**A**) | (**DCV**) | (**DSV**) | (**FMCV**) | (**FMSV**) | (**LFCV**) | (**LFSV**) | (**MD**))\* (Signo agrupación “}” ) | if(valor3){  call moverMotor()!  }else{    } | |
| **Estructura de control Loop (L)** | | |
| (**loop**) (Signo agrupación “{” ) ((**MS**) | (**MIC**) | (**MIL**) | (**A**) | (**DCV**) | (**DSV**) | (**FMCV**) | (**FMSV**) | (**LFCV**) | (**LFSV** | **IR** | **IL2V** | **IL1V** | **ER** | **EL2V** | **EL1V**) | (**MD**))\* (Signo agrupación “}” ) | loop{  if(valor3 and valor4){  call moverMotor()!  }else{    }  call Distancia = proximitySensor.distance()!    } | |