**ANÁLISIS SEMÁNTICO**

El análisis semántico es la tercera fase de un compilador y se encarga de darle sentido a las estructuras generadas, estas estructuras se les da sentido evaluando los componentes léxicos por las que están hechas, por ende necesitamos los componentes léxicos generados, los tipos de datos que existen con su debido limite, las estructuras que tienen sentido acorde a sus componentes léxicos y con ello identificar y generar los posibles errores que pueden generar dichas estructuras, por ende los elementos anteriormente mencionados se definen para este compilador de la siguiente manera.

**COMPONENTES LÉXICOS DEL COMPILADOR**

Los componentes léxicos son aquellas cadenas de caracteres que concuerdan con un patrón, dicho patrón describe la forma en cómo se forma dicho componente léxico, por lo tanto los componentes léxicos de este compilador estarán formados por el lenguaje del mismo u otros componentes léxicos, y los componentes léxicos se muestran en la siguiente tabla:

|  |  |
| --- | --- |
| **COMPONENTE LÉXICO** | **PATRÓN** |
| **NUMERO\_ENTERO** | ({Digito})({Digito})\* |
| **NUMERO\_REAL** | ({Digito})+.({Digito})({Digito})\* |
| **CONDICIONAL** | (true | false) |
| **IDENTIFICADOR\_CADENA** | ({Comilla})({Letra} | {Dígito})({Letra} | {Dígito} | {Símbolo})\*({Comilla}) |
| **IDENTIFICADOR** | ({Letra})({Letra} | {Digito})\* |
| **VALORES** | (NUMERO\_ENTERO | NUMERO\_REAL | IDENTIFICADOR\_CADENA | CONDICIONAL) |
| **PUERTO** | (port\_A0 | port\_A1 | port\_A2 | port\_A3 | port\_A4 | port\_A5 | port\_B0 | port\_B1 | port\_B2 | port\_B3 | port\_B4 | port\_B5 | port\_B6 | port\_B7 | port\_C2 | port\_C3 | port\_C4 | port\_C5 | port\_C6 | port\_C7 | port\_D0 | port\_D1 | port\_D2 | port\_D3 | port\_D4 | port\_D5 | port\_D6 | port\_D7) |
| **TIPO\_PUERTO** | (proximity | temperature | LED | LCD | button | motor) |
| **TIPO\_DATO** | (int | string | decimal | boolean) |
| **MOTOR** | (move | restart | start) |
| **METODO\_SENSOR** | (distance | state | time | degree) |
| **ESTRUCTURA\_DE\_CONTROL** | (function | begin | loop | if | else) |
| **PALABRA\_RESERVADA** | (ligther | delay | operation | call | return | print | console | turn\_off) |
| **DECLARACION** | (var | const) |
| **OPERADOR\_LOGICO** | (and | or | not) |
| **OPERADOR\_RELACIONAL** | (> | < | >= | <= | ==) |
| **OPERADOR\_ARITMETICO** | (+ | - | \* | /) |
| **OPERADOR\_ASIGNACION** | (=) |
| **SIGNO\_AGRUPACION** | ( { | } | ( | )) |
| **SIGNO\_PUNTUACION** | (, | .) |
| **FIN\_DE\_LINEA** | (!) |

**TIPOS DE DATOS DEL COMPILADOR**

Los tipos de datos son aquellos valores que pueden tener las variables en este compilador, donde también cuentan con límite y espacio en memoria, por ende los tipos de datos del compilador que son aceptados están en la siguiente tabla:

|  |  |  |
| --- | --- | --- |
| **TIPO DE DATO** | **LIMITE** | **ESPACIO EN MEMORIA** |
| **int** | -65536 - 65535 | 3 Byte |
| **string** | Cadena de 30 caracteres | 30 Byte |
| **decimal** | 1.4E-45 – 3.4028235E38 | 4 Byte |
| **boolean** | 0 o 1 (true o false) | 1 Byte |
| **puerto** | proximity, temperature, LED, LED\_RGB, LCD, button, motor | 3-11 Byte |
| **sensor** | distance, state, time, degree | 4-8 Byte |

**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()!    } | |

**ERRORES SEMÁNTICOS DEL COMPILADOR**

Los errores semánticos son aquellas estructuras que cuentan con valores incorrectos entre sí, no hay coherencia, los errores son los siguientes:

|  |  |
| --- | --- |
| **NOMBRE DEL ERROR** | **DESCRIPCIÓN** |
| **Error\_semántico\_0** | La variable ya ha sido declarada en la declaración de puerto |
| **Error\_semántico\_1** | El puerto ya está ocupado por un componente electrónico en la declaración de puerto |
| **Error\_semántico\_2** | La variable ya ha sido declarada en la declaración |
| **Error\_semántico\_3** | El tipo de dato de la declaración no corresponde con el valor asignado en la declaración |
| **Error\_semántico\_4** | El valor asignado en la declaración no se encuentra en rango |
| **Error\_semántico\_5** | La variable no ha sido declarada en la asignación |
| **Error\_semántico\_6** | La variable es de tipo const en la asignación y no se puede modificar su valor |
| **Error\_semántico\_7** | El tipo de dato de la variable no corresponde con el valor asignado en la asignación |
| **Error\_semántico\_8** | El valor asignado en la asignación no se encuentra en rango |
| **Error\_semántico\_9** | La variable 1 no está asociada a un componente electrónico de tipo motor en la función del motor |
| **Error\_semántico\_10** | La variable 2 no está asociada a un componente electrónico de tipo button en la función del motor |
| **Error\_semántico\_11** | La función del motor move no tiene un valor |
| **Error\_semántico\_12** | El valor no es de tipo entero en la función del motor |
| **Error\_semántico\_13** | El valor aceptable es 180 en la función del motor |
| **Error\_semántico\_14** | La función del motor start o restart tiene un valor |
| **Error\_semántico\_15** | La variable 1 no ha sido declarada en la función del motor |
| **Error\_semántico\_16** | La variable 2 no ha sido declarada en la función del motor |
| **Error\_semántico\_17** | El valor no es de tipo de dato string en el método de la impresora |
| **Error\_semántico\_18** | El valor no se encuentra en rango en el método de la impresora |
| **Error\_semántico\_19** | La variable no es de tipo de dato string en el método de la impresora |
| **Error\_semántico\_20** | La variable no ha sido declarada en el método de la impresora |
| **Error\_semántico\_21** | La variable 1 no está asociado a un componente electrónico de tipo LCD en el método de la impresora |
| **Error\_semántico\_22** | La variable 1 no ha sido declarada en el método de la impresora |
| **Error\_semántico\_23** | La variable 2 no es de tipo de dato string en el método de la impresora |
| **Error\_semántico\_24** | La variable 2 no ha sido declarada en el método de la impresora |
| **Error\_semántico\_25** | La variable 1 no es de tipo de dato string en el método del sensor |
| **Error\_semántico\_26** | La variable 1 no ha sido declarada en el método del sensor |
| **Error\_semántico\_27** | La variable 1 es de tipo const y no es posible modificar su valor en el método del sensor |
| **Error\_semántico\_28** | La variable 2 no está asociado a un componente electrónico en el método del sensor |
| **Error\_semántico\_29** | La variable 2 no ha sido declarada en el método del sensor |
| **Error\_semántico\_30** | El método del sensor no es el adecuado al componente electrónico en el método del sensor |
| **Error\_semántico\_31** | La variable no es de tipo de dato int en el método delay |
| **Error\_semántico\_32** | La variable no ha sido declarada en el método delay |
| **Error\_semántico\_33** | El valor no es de tipo de dato int en el método delay |
| **Error\_semántico\_34** | El valor no se encuentra en rango en el método delay |
| **Error\_semántico\_35** | La variable 1 no está asociado a un componente electrónico de tipo LED en el método encender |
| **Error\_semántico\_36** | La variable 1 no ha sido declarada en el método encender |
| **Error\_semántico\_37** | La variable 2 no está asociado a un componente electrónico de tipo button en el método encender |
| **Error\_semántico\_38** | La variable 2 no ha sido declarada en el método encender |
| **Error\_semántico\_39** | La variable 1 no está asociado a un componente electrónico de tipo LED en el método apagar |
| **Error\_semántico\_40** | La variable 1 no ha sido declarada en el método apagar |
| **Error\_semántico\_41** | La variable 2 no está asociado a un componente electrónico de tipo button en el método apagar |
| **Error\_semántico\_42** | La variable 2 no ha sido declarada en el método apagar |
| **Error\_semántico\_43** | La variable 1 de la condición no es de tipo de dato int o decimal en el if |
| **Error\_semántico\_44** | La variable 1 de la condición no ha sido declarada en el if |
| **Error\_semántico\_45** | El valor 1 de la condición no es de tipo de dato int o decimal en el if |
| **Error\_semántico\_46** | El valor 1 de la condición no se encuentra en rango en el if |
| **Error\_semántico\_47** | La variable 2 de la condición no es de tipo de dato int o decimal en el if |
| **Error\_semántico\_48** | La variable 2 de la condición no ha sido declarada en el if |
| **Error\_semántico\_49** | El valor 2 de la condición no es de tipo de dato int o decimal en el if |
| **Error\_semántico\_50** | El valor 2 de la condición no se encuentra en rango en el if |
| **Error\_semántico\_51** | El valor o la variable 1 y el valor o la variable 2 no son del mismo tipo de dato |
| **Error\_semántico\_52** | La variable 1 de la condición no es de tipo de dato boolean en el if |
| **Error\_semántico\_53** | El valor 1 de la condición no es de tipo de dato boolean en el if |
| **Error\_semántico\_54** | La variable 2 de la condición no es de tipo de dato boolean en el if |
| **Error\_semántico\_55** | El valor 2 de la condición no es de tipo de dato boolean en el if |
| **Error\_semántico\_56** | La variable de la condición no es de tipo de dato boolean en el if |
| **Error\_semántico\_57** | La variable de la condición no ha sido declarada en el if |
| **Error\_semántico\_58** | El valor de la condición no es de tipo de dato boolean en el if |