CE1108 –Compiladores e Intérpretes

Writing machine

Definición del Lenguaje de Alto Nivel

Profesor:
Marco Hernández Vásquez

Estudiante:
Andrés Molina Redondo | 2020129522
Luis Alfredo González Sánchez | 2021024482
Mariana Rojas Rojas | 2020076936
Jose Maria Vindas Ortiz | 2022209471

Segundo Semestre

Año 2024

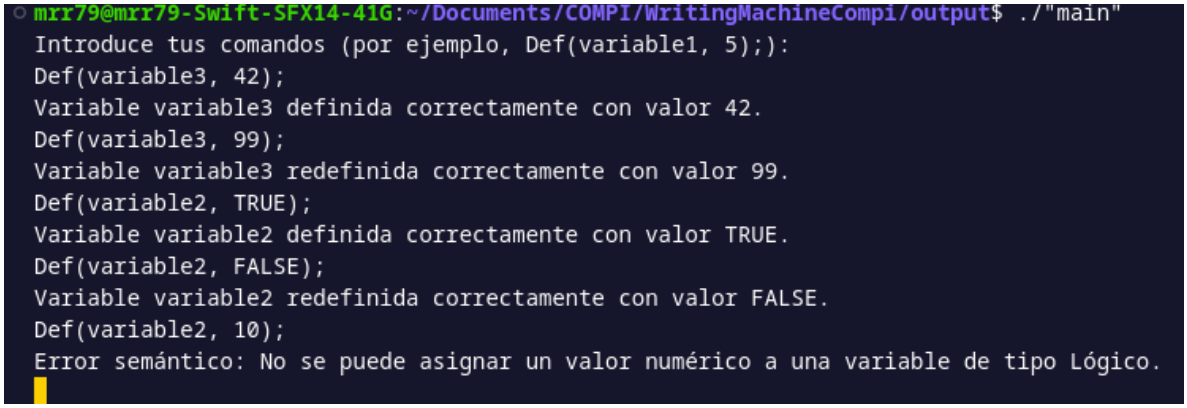
Sintaxis del lenguaje

Parte importante de este proyecto es poder ser capaces de definir una sintaxis la cual va a ser utilizada para dictarle las indicaciones al compilador, en esta ocasión un requisito importante es partir de la base de la siguiente gramática.

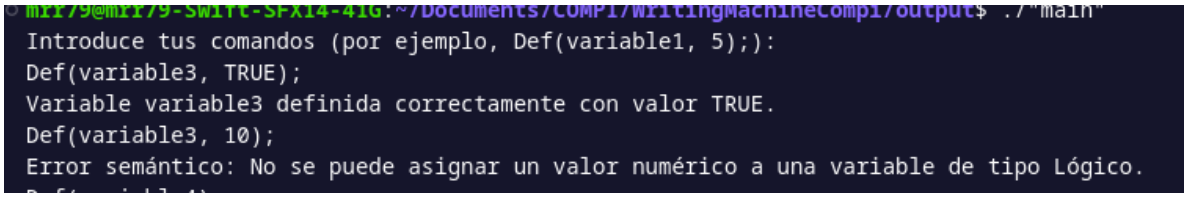
- **Def (nombre_variable, valor);**

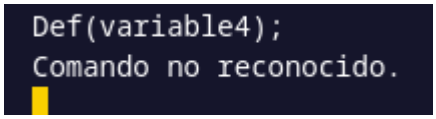
  Se usa para definir una variable, el valor que se le asigne depende del tipo de dato que se quiera guardar en la variable. Ésta puede guardar los siguientes tipos de datos, números y lógicos. En el caso ser numero solamente se acepta enteros positivos y para los lógicos únicamente TRUE y FALSE

  Ejemplo de implementación



  Ejemplo de errores semánticos





- **Put (identifica, n);**

  Altera el valor actual de la variable por el dato colocado en la instrucción. n es el valor para usar. Acepta no solamente un valor sino también una operación, por tanto, el valor de n puede ser el resultado de una operación.

  Ejemplo de implementación

```
mrr79@mrr79-Swift-SFX14-41G:~/Documents/COMPI/WritingMachineCompi/output$ ./"main"
Introduce tus comandos (por ejemplo, Def(variable1, 5); o Put(variable1, 100);):
Def(variable1, 42);
Variable variable1 definida correctamente con valor 42.
Put(variable1, 100);
Variable variable1 actualizada correctamente con valor 100.
Def(variable2, 200);
Variable variable2 definida correctamente con valor 200.
Put(variable2, Substr(300, 100));
Variable variable2 actualizada con el resultado de la operación: 200.
```

- **En caso de necesitar aumentar el valor de una variable se tiene las siguientes dos opciones**
  - **Add (N1);**

    El incremento será valor identificador + 1.

  - **Add(N1,N2);**

    En caso de que se tengan una variable y un valor. Se incrementa N1 en N2 veces. N2 puede ser una operación matemática o el valor de otra variable.

N1 siempre debe ser una variable pues es la que sufre el incremento. En caso de no ser una variable, debe generar un error.

Ejemplo de implementación

```
mrr79@mrr79-Swift-SFX14-41G:~/Documents/COMPI/WritingMachineCompi/output$ ./"main"
Introduce tus comandos (por ejemplo, Def(variable1, 5); o Put(variable1, 100); o Add(variable1);)
:
Def(var1, 10);
Variable var1 definida correctamente con valor 10.
Def(var2, 5);
Variable var2 definida correctamente con valor 5.
Add(var1);
Variable var1 incrementada correctamente. Nuevo valor: 11.
Add(var1, var2);
Variable var1 incrementada correctamente. Nuevo valor: 16.
Add(var1, 3);
Variable var1 incrementada correctamente. Nuevo valor: 19.
```

Ejemplos de errores semánticos

```
Add(var_inexistente);
Error: La variable var_inexistente no existe.
Def(var_logica, TRUE);
Variable var_logica definida correctamente con valor TRUE.
Add(var_logica);
Error semántico: Solo se pueden incrementar variables numéricas.
Add(var1, var_inexistente)
Comando no reconocido.
Add(var1, var_inexistente);
Error: La variable var_inexistente no existe.
Def(var_logica, FALSE);
Variable var_logica redefinida correctamente con valor FALSE.
Add(var1, var_logica);
Error semántico: El incremento debe ser un número o una variable numérica.

Comando no reconocido.
Add var1;
Comando no reconocido.
Add(var1, -5);
Comando no reconocido.
Add(var1, Substr(10, TRUE));
Comando no reconocido.
```

- **ContinueUp n;**

  Esta funcionalidad sirve para moverse n cantidad de unidades hacia arriba (eje y). El movimiento es de una posición a otra posición hasta alcanzar n. Es importante aclarar que n puede ser una constante numérica, una operación matemática u otra variable ya inicializada.

  Ejemplo de uso

```
andres@andres-Desktop:~/Compi/Proyecto/WritingMachineCompi$ ./main
PosX: 1, PosY: 1, Lapiz: 0
Introduce tus comandos (por ejemplo, Def(variable1, 5); o Put(variable1, 100); o Add(variable1);):
Def(variable1, 5);
Variable variable1 definida correctamente con valor 5.
ContinueUp(5);
Moviéndose a la posición Y: 6
ContinueUp(variable1);
Moviéndose a la posición Y: 11
ContinueUp(2*2);
Moviéndose a la posición Y: 15
```

  Ejemplo de errores

```
ContinueUp(TRUE);
Error: Variable no encontrada: TRUE
Moviéndose a la posición Y: 15
COntinueUp(5,5);
Comando no reconocido.
ContinueUp(5,5);
Comando no reconocido.
ContinueUp(5.5);
Comando no reconocido.
```

- **ContinueDown n;**

  Esta funcionalidad sirve para moverse n cantidad de unidades hacia abajo (eje y). El movimiento es de una posición a otra posición hasta alcanzar n. Es importante aclarar que n puede ser una constante numérica, una operación matemática u otra variable ya inicializada.

  Ejemplo de uso

```
andres@andres-Desktop:~/Compi/Proyecto/WritingMachineCompi$ ./main
PosX: 1, PosY: 1, Lapiz: 0, Color: 1
Introduce tus comandos (por ejemplo, Def(variable1, 5); o Put(variable1, 100); o Add(variable1);):
Def(variable1, 5);
Variable variable1 definida correctamente con valor 5.
ContinueDown(5);
Moviéndose a la posición Y: 6
ContinueDown(2*2);
Moviéndose a la posición Y: 10
ContinueDown(variable1);
Moviéndose a la posición Y: 15
```

  Ejemplo de errores

```
ContinueDown(TRUE);
Error: Variable no encontrada: TRUE
Moviéndose a la posición Y: 15
ContinueDown();
Comando no reconocido.
ContinueDown(2.2);
Comando no reconocido.
```

- **ContinueRight n;**

  Esta funcionalidad sirve para moverse n cantidad de unidades hacia la derecha (eje x). El movimiento es de una posición a otra posición hasta alcanzar n. Es importante aclarar que n puede ser una constante numérica, una operación matemática u otra variable ya inicializada.

  Ejemplo de uso

```
andres@andres-Desktop:~/Compi/Proyecto/WritingMachineCompi$ ./main
PosX: 1, PosY: 1, Lapiz: 0, Color: 1
Introduce tus comandos (por ejemplo, Def(variable1, 5); o Put(variable1, 100); o Add(variable1);):
Def(variable1, 5);
Variable variable1 definida correctamente con valor 5.
ContinueRight(5);
Moviéndose a la posición X: 6
ContinueRight(2*2);
Moviéndose a la posición X: 10
ContinueRight(variable1);
Moviéndose a la posición X: 15
```

Ejemplo de errores

```
ContinueRight(TRUE);
Error: Variable no encontrada: TRUE
Moviéndose a la posición X: 15
ContinueRight();
Comando no reconocido.
ContinueRight(2.2);
Comando no reconocido.
ContinueRight(200);
Error: El valor ingresado excede el limite de la hoja
```

- **ContinueLeft n;**

  Esta funcionalidad sirve para moverse n cantidad de unidades hacia la izquierda (eje x). El movimiento es de una posición a otra posición hasta alcanzar n. Es importante aclarar que n puede ser una constante numérica, una operación matemática u otra variable ya inicializada.

  Ejemplo de uso

```
andres@andres-Desktop:~/Compi/Proyecto/WritingMachineCompi$ ./main
PosX: 1, PosY: 1, Lapiz: 0, Color: 1
Introduce tus comandos (por ejemplo, Def(variable1, 5); o Put(variable1, 100); o Add(variable1);):
ContinueRight(20);
Moviéndose a la posición X: 21
Def(variable1, 5);
Variable variable1 definida correctamente con valor 5.
ContinueLeft(5);
Moviéndose a la posición X: 16
ContinueLeft(2*2);
Moviéndose a la posición X: 12
ContinueLeft(variable1);
Moviéndose a la posición X: 7
```

  Ejemplo de errores

```
ContinueLeft(TRUE);
Error: Variable no encontrada: TRUE
Moviéndose a la posición X: 7
ContinueLeft();
Comando no reconocido.
ContinueLeft(2.2);
Comando no reconocido.
ContinueLeft(200);
Error: El valor ingresado excede el limite de la hoja7
```

- **Pos(X,Y);**

Sirve para colocar el lapicero en la posición de las coordenadas X, Y. En este caso X y Y pueden ser constante numérica, operación matemática u otra variable ya inicializada.

Ejemplo de uso

```
andres@andres-Desktop:~/Compi/Proyecto/WritingMachineCompi$ ./main
PosX: 1, PosY: 1, Lapiz: 0, Color: 1
Introduce tus comandos (por ejemplo, Def(variable1, 5); o Put(variable1, 100); o Add(variable1);):
Pos(100,0);
Lapicero colocado en la posición X: 100, Y: 0
Pos(25,25);
Lapicero colocado en la posición X: 25, Y: 25
Def(variable1, 5);
Variable variable1 definida correctamente con valor 5.
Pos(variable1,6);
Lapicero colocado en la posición X: 5, Y: 6
Pos(22,variable1);
Lapicero colocado en la posición X: 22, Y: 5
```

- **PosX n;**

Coloca el lapicero en la coordenada X especificada. Un aspecto importante es que se debe recordar y respetar la coordenada Y en la cual se encuentra actualmente el lapicero. Para este caso n puede ser una constante numérica, una operación matemática u otra variable ya inicializada.

Ejemplo de uso

```
andres@andres-Desktop:~/Compi/Proyecto/WritingMachineCompi$ ./main
PosX: 1, PosY: 1, Lapiz: 0, Color: 1
Introduce tus comandos (por ejemplo, Def(variable1, 5); o Put(variable1, 100); o Add(variable1);):
PosX 20;
Lapicero colocado en la posición X: 20
PosX 2*2;
Lapicero colocado en la posición X: 4
Def(variable1, 5);
Variable variable1 definida correctamente con valor 5.
PosX variable1;
Lapicero colocado en la posición X: 5
```

Ejemplo de errores

```
PosX 120;
Error: La nueva posición X del lápiz excede los límites
```

- **PosY n;**

Se usa para colocar el lapicero en la coordenada Y especificada. Para este caso n puede ser una constante numérica, una operación matemática u otra variable ya inicializada. Al igual que en la función anterior pero esta vez para X se debe recordar y respetar la coordenada X en la cual se encuentra actualmente el lapicero.

Ejemplo de uso

```
PosY 20;
Lapicero colocado en la posición Y: 20
PosY variable1;
Lapicero colocado en la posición Y: 5
PosY 2*2;
Lapicero colocado en la posición Y: 4
```

Ejemplo de errores

```
PosY 120;
Error: La nueva posición Y del lápiz excede los límites:
```

- **UseColor valor;**

  Para este proyecto se debe disponer unicamente de dos posibles colores a usar los valores van a ser 1 en caso de que se quiera usar el color negro o 2 en caso de que se quiera usar el color rojo. En caso de no ser asignado, se usará el 1 por defecto.

  Ejemplo de uso

```
andres@andres-Desktop:~/Compi/Proyecto/WritingMachineCompi$ ./main
PosX: 1, PosY: 1, Lapiz: 0, Color: 1
Introduce tus comandos (por ejemplo, Def(variable1, 5); o Put(variable1, 100); o Add(variable1);):
UseColor 2;
Se ha asignado el valor de 2 al color de lápiz
UseColor 1;
Se ha asignado el valor de 1 al color de lápiz
```

  Ejemplo de errores

```
UseColor 0;
Comando no reconocido.
UseColor 3;
Comando no reconocido.
UseColor True;
Comando no reconocido.
```

- **Down;**

  Se usa para indicar que se coloque el lapicero sobre la superficie para estar listo para escribir. El evento se hace sobre el compartimiento asignado en “UseColor” por tanto no se debe hacer ninguna otra especificación del compartimiento a usar.

  Ejemplo de uso

```
Down;
El lapiz se ha pegado a la superficie
Down;
El lapiz ya esta pegado a la superficie
```

- **Up;**

  Sirve para que el lapicero sea retirado de la superficie para que no se pueda seguir escribiendo. El evento se realiza en el compartimiento asignado en “UseColor” por tanto no se debe hacer ninguna otra especificación del compartimiento a usar.

  Ejemplo de uso

```
Up;
El lapiz se ha despegado de la superficie
Up;
El lapiz ya esta despegado de la superficie
```

- **Beginning;**

  Esta funcionalidad sirve para Coloca el lápiz al inicio, esto quiere decir en la posición [1,1]

  Ejemplo de uso

```
Pos(63,87);
Lapicero colocado en la posición X: 63, Y: 87
Beginning;
El lapiz se ha colocado en la posicion inicial
```

- **FOR variable(Min to Max) LOOP**
  [sentencias]
  END LOOP;

  La sentencia FOR-LOOP permite especificar un rango de números enteros, finalmente ejecuta una secuencia de instrucciones para cada número entero dentro del cuerpo de la sentencia. Variable es un identificador del valor del rango en cada iteración. No debe existir un identificador con ese nombre previamente.

  Es importante tomar encuenta que:

  - Min es un entero en el cual iniciará la sentencia de control.
  - Max es un entero en el cual finalizará la sentencia de control.
  - Max debe ser mayor a Min sino se debe generar un error.

  Ejecuta todas las instrucciones que se encuentran en el cuerpo (Ordenes) por n cantidad de veces de acuerdo con la diferencia entre Max-Min.

Ejemplo de uso

Ejemplo de errores

- **Case variable**
  When Valor Then
  [instrucciones]
  When Valor Then
  [instrucciones]
  …
  End Case ;
  O bien
  Case variable
  When Valor Then
  [instrucciones]
  When Valor Then
  [instrucciones]
  …
  Else
  [instrucciones]
  End Case;

  Consulta el valor de la variable, y dependiendo de ese valor ejecuta el cuerpo de instrucciones
  asignado.

  Para esta función se debe destacar que:
  - Puede tener un número ilimitado de “When”
  - El Else es opcional
  - El valor de la variable puede ser un valor numérico o booleano.

  Ejemplo de uso

  Ejemplo de errores

- **Repeat**
  [instrucciones]
  Until
  [condición];

  Repite la lista de instrucciones tantas veces hasta que se cumpla la condición. Primero ejecuta el conjunto de instrucciones, de esta forma se asegura que las

instrucciones se ejecutan al menos una vez antes de comprobar la condición. En caso de no cumplirse, se vuelve a ejecutar el conjunto de instrucciones.

Ejemplo de uso

Ejemplo de errores

- **While [condición]**
  [instrucciones]
  Whend;

  Repite la lista de instrucciones tanta veces hasta que se cumpla la condición. Si la condición expresada NO se cumple no se ejecutan las instrucciones ni una sola vez.

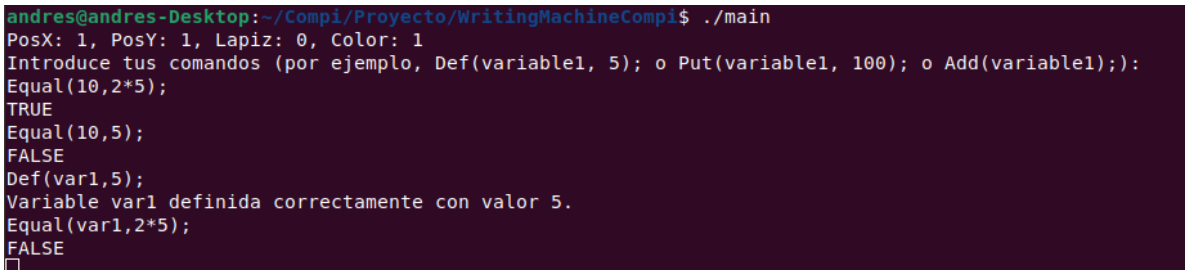
  Ejemplo de uso

  Ejemplo de errores

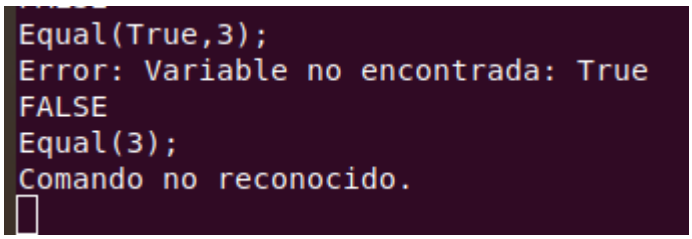
- **Equal(N1,N2)**

  Devuelve TRUE si N1 y N2 son iguales, de lo contario devuelve FALSO. Solo puede comparar valores, operaciones y/o variables.

  Ejemplo de uso:



  Ejemplo de errores:



- **And(N1,N2)**

  Esta operación devuelve TRUE si tanto la condición N1 como N2 son ciertos. Solo puede comparar valores, operaciones y/o variables.

  Ejemplo de uso

```
And(10>2,2>5);
FALSE
```

Ejemplo de errores

- **Or(N1, N2)**

  Devuelve True si al menos una de las condiciones es cierta. Solo puede comparar valores, operaciones y/o variables.

  Ejemplo de uso

```
Or(10>2,2>5);
TRUE
```

- **Greater(N1,N2)**

  Devuelve cierto si N1 es mayor a N2. Solo puede comparar valores, operaciones y/o variables.

  Ejemplo de uso

```
Greater(10, 2*5);
FALSE
Def(var1,5);
Variable var1 definida correctamente con valor 5.
Greater(var1,2*5);
FALSE
```

- **Smaller(N1, N2)**

  Devuelve cierto si N1 es menor a N2. Solo puede comparar valores, operaciones y/o variables.

  Ejemplo de uso

```
Smaller(10, 2*5);
FALSE
Smaller(2,2*5);
TRUE
Def(var1,2);
Variable var1 definida correctamente con valor 2.
Smaller(var1,2*5);
TRUE
```

- **Substr(N1, N2)**

  Esta función siempre retornará un valor.

  N1 puede ser una variable o un valor numérico, y a su valor es "restado" por el valor retornado en N2.

  N2 puede ser números, variables u operaciones.

  Es importante que N1 siempre debe ser mayor o igual a N2, de lo contrario generará un error.

  Ejemplo de uso

```
Substr(100, 45);
El resultado es: 55
Def(var2,100);
Variable var2 definida correctamente con valor 100.
Substr(var2,45);
El resultado es: 55
Substr(var2,var1);
El resultado es: 98
```

- **Random(n)**

  Se usa para generar un número aleatorio comprendido entre 0 y n. Este caso "n" puede ser una variable.

  Ejemplo de uso

```
Random(360);
Número aleatorio entre 0 y 360: 285
Random(80);
Número aleatorio entre 0 y 80: 18
Random(var2);
Número aleatorio entre 0 y var2: 49
```

- **Mult(N1, N2)**

  Esta función siempre retornará un valor.

  N1 puede ser una variable o un valor numérico, y a su valor es “multiplicado” por el valor retornado en N2.

  N2 puede ser números, variables u operaciones.

  Ejemplo de uso

```
Mult(2,5);
El resultado es: 10
Mult(5,5);
El resultado es: 25
Mult(2,2);
El resultado es: 4
```

- **Div (N1, N2)**

  Esta función siempre retornará un valor.

  N1 puede ser una variable o un valor numérico, y a su valor es “dividido” por el valor retornado en N2.

  N2 puede ser números, variables u operaciones.

  Es importante destacar que solamente se trabajará con valores enteros, por tanto, siempre se truncará el resultado.

  Ejemplo de uso

```
Div(12, 4);
El resultado es: 3
Div(var2,4);
El resultado es: 25
```

- **Sum(N1, N2)**

  Esta función siempre retornará un valor.

  N1 puede ser una variable o un valor numérico, y a su valor es “sumado” por el valor retornado en N2.

  N2 puede ser números, variables u operaciones.

Ejemplo de uso

```
Sum(1, 2);
El resultado es: 3
Sum(20,80);
El resultado es: 100
```