Programación Profesional

Trabajo final: DSL – GraphDSL

AUS - 2022

Integrantes:

## Nicolas Agustín Sandez

## Dardo Eloy

Propuesta

El proyecto consiste en un lenguaje especifico de dominio para la generación de diagramas de grafos en ***LaTeX***. Dicho lenguaje, que lo llamaremos ***GraphDSL***, cuenta con bucles, salida por pantalla, condicionales, variables enteras y cadenas de caracteres.

Propiedades

El lenguaje soporta los tipos de datos: **Enteros** y **Strings**.

Las operaciones disponibles para enteros son:

1. Suma
2. Resta
3. División
4. Multiplicación
5. Igual que
6. Menor
7. Mayor
8. Menor o igual
9. Mayor o igual
10. Distinto
11. Casteo a string

Las operaciones disponibles para strings son:

1. Concatenación
2. Longitud
3. Casteo a entero

Las operaciones de E/S son:

1. Imprimir por pantalla

Las operaciones entre nodos son:

1. (***->***) Arista dirigida hacia la derecha.
2. (***<-***) Arista dirigida hacia la izquierda.
3. (***<->***) Arista bidireccional

***GraphDSL*** cuenta con funciones nativas que permiten el manejo de salida por pantalla, longitud de cadena, casteo entre entero y string, inserción de un nodo, seteo de aristas entre nodos y creación de grafico. A continuación, se describen y se ejemplifican:

* Len: Retorna la longitud de una cadena de caracteres.
  + Parámetros: Expresión de string
  + Retorno: Entero
  + Caso de uso:

string cadena = “Hola”;

print(str(len(cadena))) #Imprime 4

* Int: Convierte una string en entero.
  + Parámetros: Expresión de string
  + Retorno: Entero
  + Caso de uso:

string numero\_str = “1”;

int suma = 1 + int(numero\_str);

print(str(suma)) #Imprime 2

* Str: Convierte un entero en cadena de caracteres.
  + Parámetros: Expresión entera
  + Retorno: String
  + Caso de uso:

string codigo = “Codigo”;

int numero = 1;

codigo = codigo & str(numero)

* Print: Imprime por pantalla.
  + Parámetros: Expresión de string.
  + Retorno:
  + Caso de uso:

print(“Hola mundo!!”) # Imprime un saludo

El lenguaje cuenta con estructuras de control y condicionales, a continuación, se detallan:

* Condicional

**if** ( ***boolexp*** ) { ***cmdexpr*** } **else** { ***cmdexpr*** }

* + Caso de uso: La clausula ***else*** es opcional.

int edad = 21;

if (edad >= 18) {  
 print(“Puede beber!!!”)  
} else {  
 print(“No puede beber :(”)  
}

* Pass
* Bucle  
  **while** ( ***boolexp*** ) { ***cmdexpr*** }
  + Caso de uso:

int i = 0;

while (i < 10) {  
 print(“Iteracion ” & str(i));

i = i + 1  
}

**for** ( ***intexp*** to ***intexp*** ) { ***cmdexpr*** }

* + Caso de uso:

for (0 to 10) {  
 print(“Hola”)  
}

# Output:

# Hola

# ...

# Hola

Para declarar variables se utilizan las palabras reservadas ***int*** y ***string***. Ejemplos:

int numero = 15; # Definicion de un entero

string saludo = “Buenos dias”; # Definicion de una string

numero = numero \* 2; # Asignacion de un entero

saludo = “Buenas noches”; # Asignacion de una string

Las primitivas para trabajar con los grafos son, GRAPH, insert y set:

La sentencia GRAPH, indica que se creara un archivo de salida (**.pdf**) con el nombre especificado por el primer argumento ***strexp***.

**GRAPH**( ***strexp***, ***intexp*** );

**…**

**END**

La sentencia **set** permite establecer una relación entre 2 nodos (->, <-, <->).

**Set** (***strexp***) ***operador*** (***strexp***)

La sentencia **insert** permite insertar un nodo al grafo especificando el tag que será visualizada, identificador para el manejo interno y la posición que este ocupará con respecto a algún otro nodo, este último parámetro es opcional.

**Insert** (***strexp***, ***strexp***,[***above*** | ***below*** | ***right*** | ***left***]of ***strexp***)

Gramática

Sintaxis