Proyecto de Compiladores

Lienzo

Héctor Ricardo Méndez Sordia A01195770

Graciela García Díaz A01195698

3 de mayo de 2016

Ing. Elda Quiroga

_____ _____

Héctor Méndez Graciela García

Monterrey, Nuevo León, México

Documentación

a. DESCRIPCIÓN DEL PROYECTO

a.1) Visión, Objetivos y Alcance del Proyecto.

Visión

El objetivo de este proyecto es facilitarle a los jóvenes de enseñanza media el proceso de aprendizaje de la programación. Se desarrolló un lenguaje gráfico amigable y sencillo, de nombre Lienzo, que consiste de elementos propios a la programación, tales como ciclos y condiciones, pero que además cuenta con una pluma virtual con la que se puede dibujar libremente. Así, la salida será algo con lo que están familiarizados.

Los usuarios de este lenguaje serán jóvenes de prepa o secundaria que no tengan nociones de programación y que no dominen el idioma inglés. El lenguaje será parecido al español. De este modo, abrimos las oportunidades a los jóvenes hispanófonos que no dominan el idioma inglés.

Objetivo

El objetivo del lenguaje Lienzo es permitir la creación de gráficos simples mediante instrucciones sencillas de ejecutar. Lienzo será natural de alto nivel y sus palabras reservadas parecidas al español. Además, será imperativo y se ejecutará en una computadora de propósito general.

Lienzo servirá para darles oportunidad de aprender a programar a los jóvenes de secundaria que no dominan el inglés. Permitirá dibujar figuras geométricas bidimensionales sencillas, tales como cuadrados, círculos, etc.. y darles animación, lo cual hará sin duda divertido el proceso de aprendizaje.

a.2) Análisis de Requerimientos y Casos de Uso generales.

a.3) Descripción de los principales Test Cases.

Los casos de uso utilizados fueron los siguientes:

- El usuario puede compilar un programa básico, que consiste de instrucciones, redactadas correctamente de acuerdo al léxico y sintaxis del lenguaje, que contienen asignaciones de variables y operaciones aritméticas.

- El programa cuenta con condiciones que son analizadas correctamente.

a.4) Descripción del PROCESO general seguido para el desarrollo del proyecto, incluyendo Bitácoras generales y un pequeño párrafo de reflexión de cada alumno, en relación a los principales aprendizajes logrados (firmarlo).

El proyecto fue desarrollado en Github con el fin de tener un control de versiones de todo el proyecto. Cada semana se produjo un avance y se subió a la plataforma de Blackboard junto con una descripción del mismo. A continuación se muestran dichas descripciones con sus fechas:

Bitácora

3/3/16 8:36 PM

Primer Avance
Scanner y Parser en ANTLR funcionando al 100%

3/10/16 11:47 AM

Segundo Avance
Ya se modificó la gramática y se crearon las tablas de variables que validan si las variables están repetidas (incluyendo funciones). También se creó otro programa de prueba, el primero sí funciona y el segundo no. El directorio de funciones ya está funcionando.

3/20/16 9:42 PM

Tercer Avance
Cuadruplos funcionando al 100% con expresiones, declaraciones, asignaciones, lectura, escritura
Funciones son capaces de marcar errores si los argumentos son incorrectos.
Cubo semántico completado.

4/1/16 6:35 PM

Cuarto Avance
En este avance implementamos los cuádruplos de IF, IF ELSE y WHILE.
También arreglamos algunas cosas de la implementación, como los retornos de las funciones y estilo de sintaxis.

4/9/16 10:19 PM

Quinto Avance
Cuádruplos de funciones implementadas y funcionando al 100%.

4/16/16 10:26 PM

Sexto Avance
Fase 1 de la máquina virtual ya quedó implementada. Se traducen de cuádruplos a código meta las siguientes cosas: operaciones aritméticas, booleanas y secuenciales.

4/24/16 8:46 PM

Septimo Avance

Fase 2 de la máquina virtual: ciclos y condicionales, print, write, read, draw with Turtle

Código fuente: soporta también todo tipo de funciones y arreglos

Reflexiones Individuales

Héctor Ricardo

Graciela

b. DESCRIPCIÓN DEL LENGUAJE:

b.1) Nombre del lenguaje.

Lienzo

b.2) Descripción genérica de las principales características del lenguaje (en forma narrativa).

Lienzo es un lenguaje de programación basado en el idioma español que puede ser utilizado como un lenguaje de programación básico, con condiciones, ciclos y funciones, pero el mismo tiempo puede utilizarse para dibujar sobre un lienzo.

Principales características semánticas

Los tipos que maneja Lienzo son: numérico, mensaje, Las operaciones aritméticas únicamente podrán realizarse con expresiones de tipo numéricas.

Habrá jerarquía de operaciones. Primero se evaluarán los paréntesis, luego las multiplicaciones y divisiones de izquierda a derecha, y finalmente las sumas y restas de izquierda a derecha.

Las funciones no podrán tener el mismo nombre que las variables.

Si una hay un parámetro que se haya pasado por referencia, entonces el argumento debe estar ligado a una zona de memoria, es decir, debe ser forzosamente una variable.

Solamente puede haber arreglos de figuras. No puede haber arreglos de enteros o condiciones o colores dentro de la sección animación.

b.3) Descripción de los errores que pueden ocurrir, tanto en compilación como en ejecución.

Los errores que pueden ocurrir en compilación son los siguientes:

- No se encontró Python
- No se encontró Antlr
- No se encontró el archivo a compilar

Los errores que pueden ocurrir durante la ejecución son los siguientes:

- Errores que notificará el compilador

c. DESCRIPCIÓN DEL COMPILADOR:

c.1) Equipo de cómputo, lenguaje y utilerías especiales usadas en el desarrollo del proyecto.

Para desarrollar este proyecto se utilizó un equipo con Windows 10, Python y Antlr.

c.2) Descripción del Análisis de Léxico. Debe incluir: o Patrones de Construcción (expresados con Expresiones Regulares) de los elementos principales. o Enumeración de los "tokens" del lenguaje y su código asociado.

Los tokens asociados al lenguaje son los siguientes:

```
WS : [ \t\r\n]+ -> skip
ROJO : 'rojo'
VERDE : 'verde'
AMARILLO : 'amarillo'
AZUL : 'azul'
BLANCO : 'blanco'
NEGRO : 'negro'
MORADO : 'morado'
NARANJA : 'naranja'
CAFE : 'cafe'
GRIS : 'gris'
COLOR : 'color'
LIENZO : 'lienzo'
EQUALS : '='
TAMANO : 'tamano'
POR : 'por'
DE : 'de'
```

EN : 'en'
MOVER : 'mover'
ADELANTE : 'adelante'
ATRAS : 'atras'
GIRAR : 'girar'
DERECHA : 'derecha'
IZQUIERDA : 'izquierda'
LEVANTAR : 'levantar'
BAJAR : 'bajar'
PLUMA : 'pluma'
DIBUJO : 'dibujo'
DORMIR : 'dormir'
MIENTRAS : 'mientras'
REGRESAR: 'regresar'
QUE : 'que'
SI : 'si'
SINO : 'sino'
TEXTO : 'texto'
LEER : 'leer'
BOLEANO : 'boleano'
NUMERO : 'numero'
ESCRIBIR : 'escribir'
IMPRIMIR : 'imprimir'
NADA : 'nada'
BOOLEAN_CONSTANT : 'verdadero' | 'falso'
MODIFICABLE : 'modificable'
INTEGRAL_CONSTANT : [0-9]+
NUMERIC_CONSTANT : [0-9]+'.'[0-9]+
STRING_CONSTANT : '"' ~('"')* '"'
ID : [A-Za-z][A-Za-z0-9]*

c.3) Descripción del Análisis de Sintaxis. Debe incluir: o Gramática Formal empleada para representar las estructuras sintácticas (Sin “codificar”)

c.4) Descripción de Generación de Código Intermedio y Análisis Semántico. Debe incluir: o Código de operación y direcciones virtuales asociadas a los elementos del código. o Diagramas de Sintaxis con las acciones correspondientes. o Breve descripción de cada una de las acciones semánticas y de código (no más de 2 líneas). o Tabla de consideraciones semánticas.

c.5) Descripción detallada del proceso de Administración de Memoria usado en la compilación.

Para administrar la memoria primeramente se fueron procesando las instrucciones en lenguaje Lienzo y guardándose en cuadruplos. La clase de cuadruplos es la siguiente:

```
class Cuadruplos:
    def __init__(self):
        self.listaCuadruplos = []
        self.pilaSaltos = []

    def getCuadruplos(self):
        return self.listaCuadruplos

    def addCuadruplo(self, functionName, operator, op1, op2, t=None, generateT=True):
        if not t and generateT:
            t = TemporalRegister(functionName)
        self.listaCuadruplos.append([operator, op1, op2, t])
        return t

    def editCuadruplo(self, indice, t):
        self.listaCuadruplos[indice][3]=t

    def last(self):
        return len(self.listaCuadruplos)-1

    def pushPilaSaltos(self, indice):
        self.pilaSaltos.append(indice)

    def popPilaSaltos(self):
        return self.pilaSaltos.pop()

    def current(self):
        return len(self.listaCuadruplos)
```

La clase Cuádruplos cuenta con una lista en python donde se guarda cada cuádruplo. Los cuádruplos consisten de varios campos en donde se guardan las variables y operaciones que se utilizan en cada línea de código. Cada cuádruplo tiene asociada una dirección de memoria, y esta se maneja en la clase MemoryRegister.

d. DESCRIPCIÓN DE LA MÁQUINA VIRTUAL:

d.1) Equipo de cómputo, lenguaje y utilerías especiales usadas

Se utilizó el mismo equipo de cómputo que en el compilador. La librería adicional que se utilizó se llama Python Turtle y es para poder hacer gráficos con Python..

d.2) Descripción detallada del proceso de Administración de Memoria en ejecución (Arquitectura). Incluir: o Especificación gráfica de CADA estructura de datos usada (Memoria Local, global, etc..) o Asociación hecha entre las direcciones virtuales (compliación) y las reales (ejecución).

Para administrar la memoria en ejecución se utilizó

e). PRUEBAS DEL FUNCIONAMIENTO DEL LENGUAJE :

e.1) Incluir pruebas que "comprueben" el funcionamiento del proyecto: o Codificación de la prueba (en su lenguaje).o Resultados arrojados por la generación de código intermedio y por la ejecución.

f). LISTADOS PERFECTAMENTE DOCUMENTADOS DEL PROYECTO:

f.1) Incluir comentarios de Documentación, es decir: para cada módulo, una pequeña explicación de qué hace, qué parámetros recibe, qué genera como salida y cuáles son los módulos más importantes que hacen uso de él.

f.2) Dentro de los módulos principales se esperan comentarios de Implementación, es decir: pequeña descripción de cuál es la función de algún estatuto que sea importante de ese módulo.

Manual de Usuario

A continuación se presenta un manual que detalla las diferentes características del lenguaje Lienzo, acompañado de ejemplos donde se muestra cómo se emplea.

Sobre la instalación de herramientas requeridas

Lienzo utiliza las siguientes herramientas adicionales: Python y Antlr. A continuación se explica cómo instalar ambas

Python

Lienzo está basado en el lenguaje de programación Python. Para instalar Python se debe acudir a la siguiente liga: https://www.python.org/downloads/, seleccionar Python 3.5.1 e instalarlo en el equipo de cómputo a utilizar..

Antlr

El compilador de Lienzo está basado en la herramienta Antlr. Para instalar Antlr se debe acudir a la siguiente liga: http://www.antlr.org/download/antlr-4.5.2-complete.jar e instalar en el directorio C: del equipo de cómputo a utilizar.

Sobre la compilación

Para compilar un programa hecho en Lienzo se deben ejecutar las siguientes instrucciones:

```
java -jar C:\antlr-4.5.2-complete.jar -Dlanguage=Python3 Lienzo.g4
python script.py NombredelArchivo
```

Mi primer programa en Lienzo

A continuación se muestra cómo se escribiría el famosísimo “Hello World” en el lenguaje Lienzo.

```
Imprimir “Hello World!”
```

Convenciones de léxico

Comentarios

Este lenguaje no maneja comentarios.

Identificadores

Variables numéricas

Las variables numéricas llevan la palabra “numero” y pueden consistir de números enteros o decimales. Ejemplo: número n = 4;

Variables booleanas

Las variables booleanas llevan la palabra “boleano” y pueden llevar consigo el valor de falso o verdadero. Ejemplo: boleano e = verdadero;

Variables de texto

Las variables de texto llevan la palabra “texto” y pueden llevar consigo cualquier valor alfanumérico. Ejemplo: texto bienvenida = “Bienvenidos todos”;

Palabras reservadas

Las siguientes palabras no se pueden usar como nombres de variables o funciones porque ya están siendo utilizadas por el lenguaje Lienzo:
'rojo' , 'verde' , 'amarillo' , 'azul' , 'blanco' , 'negro' , 'morado' , 'naranja' , 'cafe' , 'gris' , 'color' , 'lienzo' , 'tamano' , 'por' , 'de' , 'en' , 'mover' , 'adelante' , 'atras' , 'girar' , 'derecha' , 'izquierda' , 'levantar' , 'bajar' , 'pluma' , 'dibujo' , 'dormir' , 'mientras' , 'regresar', 'que' , 'si' , 'sino' , 'texto' , 'leer', 'boleano' , 'numero' , 'escribir' , 'imprimir' , 'nada' , 'verdadero' , 'falso' , 'modificable'

Diseño de un programa

Los programas escritos en lenguaje lienzo tienen el siguiente orden:

Declaración de variables
Declaración de funciones
Programa principal

Las variables globales se declaran al inicio, y no es hasta después de ser declaradas que se puede hacer uso de ellas. Dentro de las funciones se sigue la misma lógica, primero se

declaran las variables a utilizar y posteriormente ya se puede hacer uso de ellas. El programa principal lel procede a las funciones, y este no está encerrado en ninguna función, son líneas de código sueltas.

Expresiones

A continuación se muestran algunos ejemplos de expresiones escritas en lenguaje Lienzo:

Aritméticas

```
variable = 1 + 4;
```

Booleanas

```
variable = falso;
```

Condiciones

Estas expresiones se pueden combinar para formar parte de condiciones. Aquí se muestran algunas de ellas:

Condicional si

```
si (variable - 4 > 0){
    variable=variable+1;
}
```

Condicional si no

```
si (variable - 4 > 0){
    variable=variable+1;
} sino{
    variable=7;
}
```

Ciclos

Las condiciones se pueden utilizar para formar ciclos que se ejecuten una y otra vez dependiendo de la condición base que contengan.

Mientras que

```
mientras que(b>5){
    b=b-1;
    imprimir b;
}
```