INFORME

Autómatas, Teoría de Lenguajes y Compiladores

Simple Keyboard Language

- Santiago Burgos (55193), Gabriel Silvatici (53122).

Indice

Objetivo del lenguaje

La idea de el lenguaje es facilitar la escritura de un programa abarcando distintos tipos de teclados. Para esto decidimos que las palabras reservadas no sean de más de tres caracteres, y que los unicos caracteres especiales sean ‘,’(coma) y ‘.’ (punto).

Elegimos estos dos porque tienen la misma disposicion tanto en teclado latinoamericano como estadounidense y no necesitan presionar otra tecla para ser usados.

Ademas vimos que en casi todos los teclados mobile de smartphones y tablets estos dos caracteres son los unicos que aparecen sin necesidad de presionar otro boton para usarlos. Facilitando mucho la escritura en este tipo de dispositivos.

Implementación

Se utilizo lex y yacc para la definición de la gramática, y se obtiene un programa en código Java, para realizar la conversión del lenguaje se usa una pila en la que se van almacenando las producciones resueltas y son siendo retiradas a medida que otras producciones tengan una dependencia de ellas, de este modo se termina teniendo una función con su traduccion a java la cual se almacena en un array de funciones para al terminar la traduccion ser copiadas al archivo out.java.

Para evitar problemas de compilacion de Java se tiene una array con los nombres de las funciones y otro con las variables, los cuales son usados para revisar que no haya funciones repetidas, que no falte la función main, y que no se declaren variables repetidas o se referencie a inexistentes.

Gramatica

Tipos de datos

- Integer
  “int ‘name’ eq 0,”

- String
  “str ‘name’ eq .Soy un string,”

La igualdad se representa con “eq”
El fin de linea se representa con “,”

Constante

“con ‘name’ eq .Soy una constante”

Operadores

“add” (+), “sub” (-), “mul” (*), “div” (/), “mod” (%), “gre” (>),
“les” (<), “and” (&&), “or” (||), “not” (!), “cmp” (==).

Programa

skl
fun ‘name’..
..
fun ‘main’...
..

Se usa “skl” para indicar inicio de programa y luego se declaran las funciones.

Llamada a funcion

“go ‘functionname’,”

Funcion

fun ‘name’ ..
..

Se usa “..” para indicar comienzo y final del cuerpo de la funcion.
Para que el programa funcione necesita una funcion main (fun main..)

Bloque Condicional

if , ‘condicion’ , ..
..

Se usa “,” para comienzo y final de la condicion, “..” para el cuerpo

Bloque while

whi , condicion , ..
..

Bloque for

for , ‘name’ eq ‘value’ to ‘target’ ..
..

Donde name puede ser una variable nueva o ya declarada, value y target pueden ser un numero o el nombre de una variable ya declarada.

For incrementa name desde value hasta target.

Ejemplo de condicion: “ , , ‘value1’ cmp 10, and ,value2 cmp 10, , “

seria equivalente a: “((‘value1’ == 10) && (‘value2’ == 10))” en C.

Salida de datos:

“put .Hola mundo,”

“put ‘name’”

Imprime el string o valor de variable en pantalla.

Entrada de datos:

“get ‘name’”

Asigna a la variable integer ‘name’ el valor de get leido de teclado. Solo se admite lecturas de integer.

Limitaciones a considerar

Las funciones son de tipo void y sin argumentos, una funcion no puede ser llamada si no fue declarada antes. Las variables no tienen un valor por defecto, deben ser seteadas al declararlas.

Se podria expandir la cantidad de tipos y explotar ventajas de las librerias de java aunque al pensarse en un lenguaje con pocos caracteres especiales habria que expandir mucho la cantidad de palabras reservadas.