Trabajo Práctico Especial

Lenguaje Aguila

Teoría de Lenguajes y Autómatas

Integrantes

Pagni, Lucio

Bacigalupo, Tomas

Pinilla, Lautaro

Marchetti, Gianfranco

Índice

Objetivo

El Lenguaje Águila está orientado a personas dando sus primeros pasos en la programación. Es un lenguaje de programación Imperativa, compilado y no tipado. Hasta esta versión se admite el uso de cadenas de caracteres y números.

Descripción

Hello World

```
inicio
imprimir “Hello World”.
fin
```

Sintaxis

La sintaxis de Águila es muy sencilla, en esta versión admite:

- Tags de `inicio` y `fin`

  En cada programa se debe indicar el inicio y el final del programa con los tags `inicio` y `fin`. A diferencia de los lenguajes que utilizan llaves u otros símbolos para delimitar bloques, estas etiquetas resultan más amigables para personas que recién comienzan en la programación.

  ```
  inicio
  ..................
  ..................
  fin
  ```

- En las asignaciones automáticamente se declara la variable y no es necesario indicar el tipo. Esto permite al programador abstraerse totalmente de los tipos de dato que tanta complicación traen.

  ```
  inicio
  text = ”this is a string”.
  num = 15.
  fin
  ```

- Se admiten las operaciones de suma, resta, multiplicacion y division.

```
inicio
num = 15.
sum = num + 5.
res = res -5.
mult = num * 5.
div = num / 5.
fin
```

- Comparaciones y condicional. Se debe siempre utilizar la palabra `si` en conjunto con la palabra `sino`

```
inicio
a = 1.
b = 2.
si a < b
imprimir “a es menor a b”.
sino
imprimir “b es menor a a”.
finsi
fin
```

- Ciclos

```
inicio
a=0.
b=5.
mientras a < b
imprimir “a es menor que b”
a = a + 1.
finciclo
fin
```

Dificultades

La principal dificultad fue armar la gramática para que nuestro lenguaje sea no tipado y se utilice como lenguaje intermedio al Lenguaje C que es fuertemente tipado.

Debido a que el lenguaje es no tipado, inspirado en el lenguaje python, existe una ambigüedad en el momento en el que se inicializa una variable o se asigna. Esto presenta una dificultad en cuanto a la claridad sintáctica del lenguaje. Es decir no es fácil distinguir si i = 1 es una inicialización o una asignación. Para ello las asignaciones de una variable preexistente debe hacerse luego de la palabra cambio. Esto es una confusión muy común entre los alumnos de programación que recién empiezan a programar. Con esta característica del lenguaje, se busca afianzar el concepto y evitar confusiones.

Futuras Extensiones

Para las próximas versiones se podrían implementar arrays y estructuras de datos más complejas.

Referencias

Se utilizaron los siguientes videos de you tube para entender con más profundidad el funcionamiento de lex y yacc.

LEX/YACC TUTORIAL PART 1 AND 2

https://www.youtube.com/watch?v=54bo1qaHAfk

https://www.youtube.com/watch?v=__-wUHG2rfM

Introduction to YACC

https://www.youtube.com/watch?v=yTXCPGAD3SQ