

Quack(!)

Un acercamiento diferente a la programación...

Propuesta de Proyecto

Compiladores

Ing. Elda Quiroga

Dr. José Icaza

David Tovar Goris
1161502

Marzo, 2013

Visión:

Para Mayo 2013 se habrá desarrollado un lenguaje de programación, que su principal función es la enseñanza e introducción a la lógica de codificación a jóvenes que les interese las ciencias computacionales, manteniendo la simplicidad en el código pero maximizando la enseñanza y facilidad de programación.

Objetivo del lenguaje:

Quack(!) Tiene como objetivos principales enseñar la lógica de codificación y resolución de problemas, así como facilitar el aprendizaje de la programación mediante código simple que genere resultados gráficos y fáciles de entender.

Principalmente, Quack(!) estará planteado para facilitar el aprendizaje de programación a alumnos de habla hispana, que les interese conocer un poco de cómo programar y sus diferentes aplicaciones.

Requerimientos del lenguaje:

A continuación se proponen los diferentes elementos del lenguaje Quack(!), tomando en cuenta sus respectivas representaciones en el lenguaje C.

Componentes del léxico:

Instrucción	Tipo	Representa	Expresión Regular
Id	-	El nombre de una variable o función	$l \cdot (l \mid d)^*$
num	Númeroico	Números, tanto enteros como flotantes, positivos y negativos.	$(+ -)?(d \cdot d^* \mid d \cdot d^*)$
texto	Texto	Cadena de caracteres, que incluye símbolos y alfanuméricos	$" \cdot ^* "$
bool	Lógico	Una expresión lógica ya sea verdadera o 1, y falsa o 0,	$(verdadero \mid falso) \mid (0 \mid 1)$

Operadores	Aritméticos
Operadores	Sintaxis
Asignación	$a = b$
Suma	$a + b$
Resta	$a - b$
Multiplicación	$a * b$
División	a / b

Operadores de	Comparación
Operadores	Sintaxis
Igual a	$a == b$
No igual a	$a != b$
Mayor que	$a > b$
Menor que	$a < b$
Mayor o igual que	$a >= b$
Menor o igual que	$a <= b$

Palabra	Representa
numero	El tipo de variable numérica
texto	El tipo de variable texto
bool	El tipo de variable boolean o lógica
haz	Equivalente a do
mientras	Equivalente a while
por	Equivalente a for
si	Equivalente a if
sino	Equivalente a else
regresa	Equivalente a return
despliega	Función básica para escribir un archivo o a consola
funcion	Define que empieza una función
var	Definición para una o más variables
dibuja	Función básica para dibujar en el lienzo
guarda	Función básica para guardar el lienzo
linea	Línea para el lienzo
circulo	Círculo a dibujar en el lienzo
triangulo	Triángulo a dibujar en el lienzo
plumaArriba	Define que no se pinte en el lienzo
plumaAbajo	Define que si se pinte en el lienzo

*NOTA: La definición de algunos grupos de símbolos y caracteres son los siguientes:

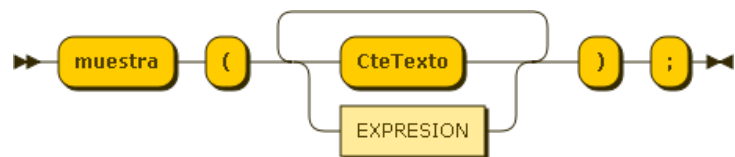
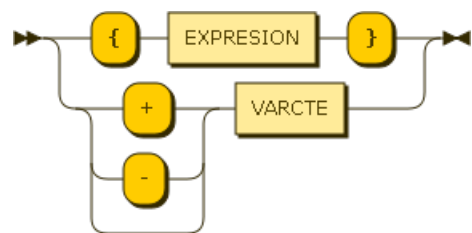
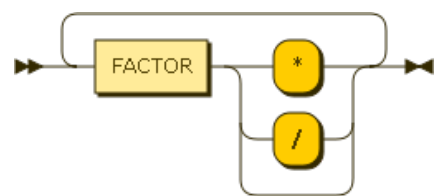
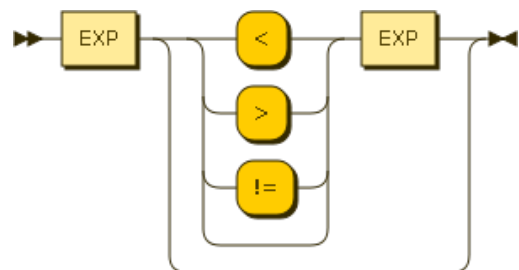
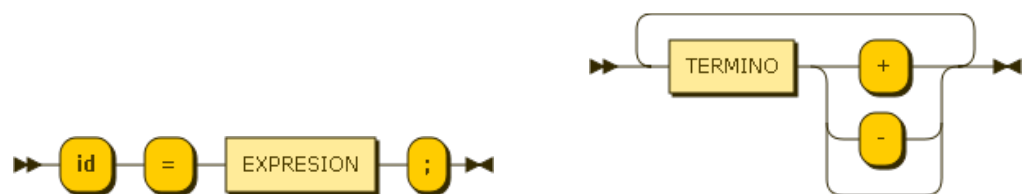
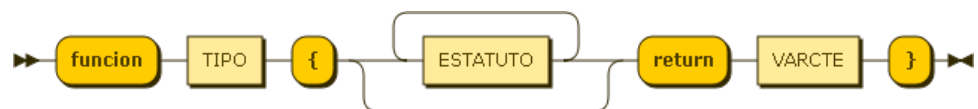
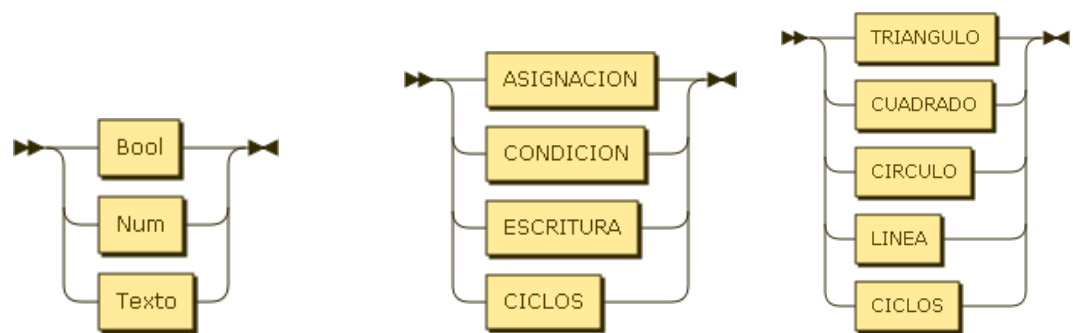
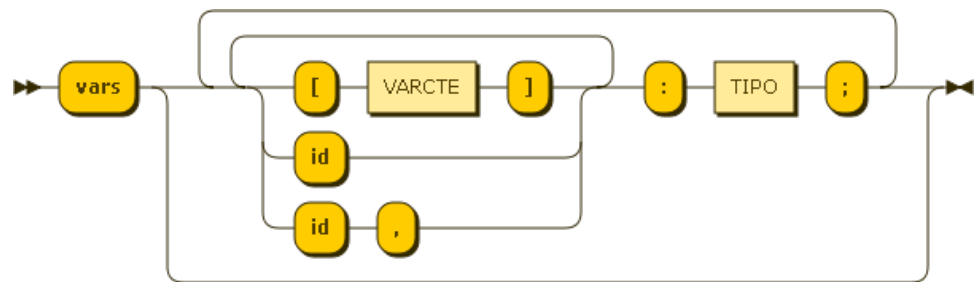
Letras = l: a-z, A-Z, _

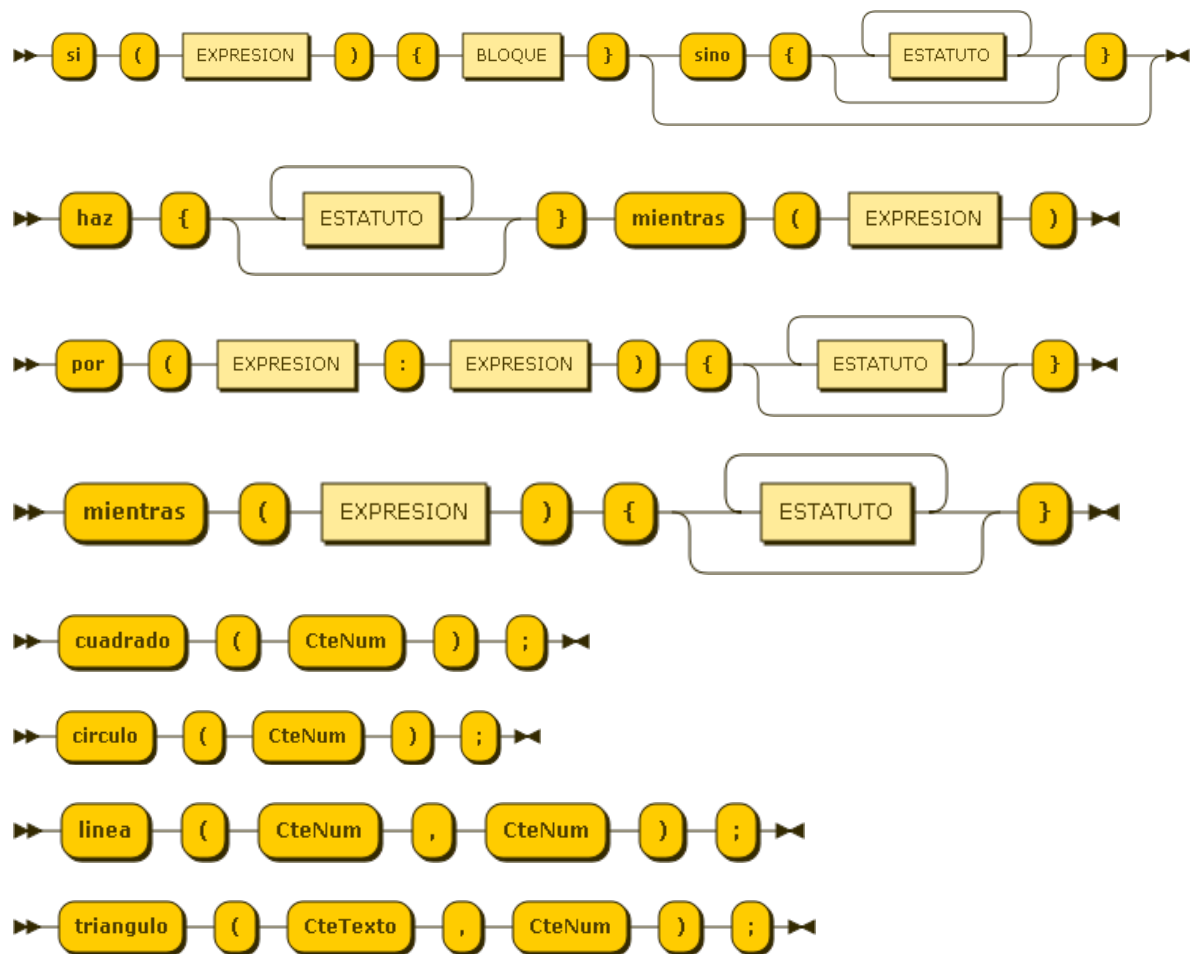
Dígitos = d: 0-9

Puntuación: ~ ! @ # % ^ & * () - + = : ; " ' < > , . ? | / \ { } []

Caracteres para espacios: espacio (" "), tabulación horizontal ("t"), salto de línea ("n")

Diagramas de sintaxis:





Principales características semánticas:

El lenguaje por sus restricciones de ser simple, no permitirá operaciones de sobrecarga de operadores ni tampoco la combinación de tipos de variables.

Descripción de las funciones especiales del lenguaje:

La principal característica del lenguaje es que, después de programar unas pocas líneas de código, se generará una salida gráfica que desplegará líneas, círculos, triángulos, o figuras libres.

Cada programa debe de tener la función `programa()` y la función `dibuja()`. En la primera se definirán las variables a trabajar y funciones y en el segundo se definirá que se quiere dibujar.

Tipos de datos del lenguaje:

Instrucción	Tipo	Límites
num	Numérico	Un número será de 64 bits como máximo o sea de – 9,223,372,036,854,775,808 a 9,223,372,036,854,775,807
texto	Texto	El texto tendrá un tamaño máximo de 2048 caracteres.
bool	Lógico	Constará de un byte, ya que sólo se necesita un '1' o un '0'

- Cabe mencionar que sólo las variables de tipo numérico podrán usar **arreglos** y serán el único tipo de dato estructurado que existirá en este lenguaje.
- Las variables están pensadas para un ambiente de 64 bits y se debe de tomar en cuenta eso para el desarrollo de las aplicaciones.
- Se espera que existan dos tipos de alcance de las variables, habrá variables globales y variables para uso específico de las funciones.

Plataforma de desarrollo:

La verificación de sintaxis y léxico será en Flex y Bison. El desarrollo principal del código para el compilador será en C/C++ y por lo tanto se usarán los IDE's XCode y Visual Studio, ya que se planea que funcione tanto en Windows como en sistemas UNIX. La máquina virtual, donde se desplegarán las líneas y dibujos, estará basada en el lenguaje Java y se usará Eclipse. También, se planea que el proyecto sea libre y por lo tanto se pondrá en un repositorio público en GitHub.

Bibliografía:

[http://msdn.microsoft.com/en-us/library/s3f49ktz\(v=vs.80\).aspx](http://msdn.microsoft.com/en-us/library/s3f49ktz(v=vs.80).aspx)

http://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B

http://en.wikipedia.org/wiki/C_variable_types_and_declarations

[http://en.wikipedia.org/wiki/C_\(programming_language\)](http://en.wikipedia.org/wiki/C_(programming_language))

[http://en.wikipedia.org/wiki/Logo_\(programming_language\)](http://en.wikipedia.org/wiki/Logo_(programming_language))