**INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES**

**CAMPUS MONTERREY**

*Compiladores: “Funpiling”*

Por:

José Miguel Rady Allende ID:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Luis Antonio Ordaz Villarreal ID: 919790

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Maestro:

Ing. Elda Guadalupe Quiroga

Fecha:

28 de Febrero del 2014

Índice

VISIÓN DEL PROYECTO 3

OBJETIVO DEL LENGUAJE 3

REQUERIMIENTOS DEL PROYECTO 4

COMPONENTES DE LÉXICO DEL LENGUAJE 4

DIVISION DE TERMINALES 4

DIAGRAMAS DE SINTAXIS 5

PRINCIPALES CARACTERÍSTICAS SEMÁNTICAS 6

EXPRESIONES 7

COMBINACIÓN DE TIPOS 8

FUNCIONES ESPECIALES DEL LENGUAJE 8

TIPOS DE DATOS 9

PLATAFORMA DE DESARROLLO 9

ANEXO 1 - PROTOTIPO 10

ANEXO 2 – CORRECCIONES 11

BIBLIOGRAFÍA 13

# VISIÓN DEL PROYECTO

Después de haber recibido la primera sesión del curso “Compiladores”, en la cual fuimos otorgados retroalimentación referente a propuestas de trabajos finales para el curso, quedamos convencidos, como equipo, de que nuestro interés se vería en la implementación de un proyecto de índole gráfico.

Coincidentemente, el semestre pasado, ambos integrantes del proyecto cursamos la asignatura de “Gráficas Computacionales”. No nos consideramos unos expertos en la materia, sin embargo observamos un área de oportunidad para profundizar un poco más en este ámbito, partiendo de los conceptos aprendidos durante el curso ya mencionado. De igual modo, vemos como un reto el poder integrar elementos gráficos con el proceso de compilación, a manera de que a partir de una fuente “gráfica” se pueda realizar un proceso de compilación y se produzca una salida gráfica también.

Nuestro lenguaje de programación no será uno con una abstracción muy elevada, pues la intención de este será el de despertar en los jóvenes la pasión por el arte de la programación, ofreciendo estructuras básicas y conceptos más digeribles. Se pretende dar una iniciación, desde un punto de vista visual, a la “lógica de la programación”.

Por último, y ya en un aspecto más personal, a través de la elaboración de este proyecto esperamos poder tener un concepto más sólido respecto al proceso que nos da cabida para ejercer nuestra profesión: la compilación. Entramos al mundo de la informática tomando por asentado la compilación, quizás hasta cierto punto abusando de ésta, pero un error sería el no adentrarse y conocer más acerca de este procedimiento. Desde puntos de vista de optimización o cierto cumplimiento de requisitos por parte de programadores, uno podría moldear su propio lenguaje de programación. Inclusive, ya desde un punto de vista no tan ingenieril, la elaboración de este proyecto podría ayudar a entender el origen de nuestros errores y ‘educarnos’ a no cometerlos más o interpretarlos. Aunado a este aspecto, también están los objetivos de aprender un nuevo lenguaje de programación y la integración y conexión entre distintos ambientes de programación.

# OBJETIVO DEL LENGUAJE

Como mencionado anteriormente, la intención del lenguaje de programación “Funpiling”, es la de proveer una iniciación al sector juvenil en el aspecto de la programación. Muchas veces la gente le rehúye a la oportunidad de aprender acerca de programación al establecerla de “muy complicada” o simplemente por desinterés. Sin embargo, en muchas ocasiones, después de contar con alguna figura de apoyo y recibir cierta inducción, usuarios terminan fascinados con este mundo de la lógica.

Un paradigma visual provee una plataforma para un mejor entendimiento del área que sea. Esto se demuestra a través de los inicios educativos de todo ser humano: imágenes son utilizadas para realizar relaciones cruciales. El cerebro humano tiende a catalogar conocimientos/términos con imágenes, de este modo creando una base para la futura interpretación de conocimientos.

De tal modo, nuestro lenguaje contará con estructuras gráficas que tendrán asociadas implementaciones de estructuras de programación, desde la declaración de variables hasta la construcción de bloques y estatutos de control. La idea inicial y 1er prototipo que se tiene de esta interfaz puede ser observado en la sección de Anexos.

# REQUERIMIENTOS DEL PROYECTO

## COMPONENTES DE LÉXICO DEL LENGUAJE

TOKENS: main, {, }, ID, =, ;, (, ), if, else, print, , , int, float, string, <, >, <>, +, -, \*, /, CTE INT, CTE FLOAT, CTE STRING, &&, ||, ~.

### DIVISION DE TERMINALES

#### LITERALES

* {
* }
* =
* ;
* (
* )
* ,
* <
* >
* +
* -
* \*
* /
* &&
* ||
* ~

#### PALABRRAS RESERVADAS

* main
* if
* else
* print
* int
* float
* string

#### ER

Conjuntos:

l = 'A'..'Z' + 'a'..'z'

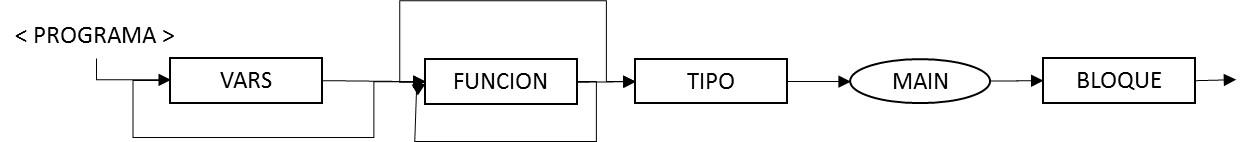
d = "0,1,2,3,4,5,6,7,8,9"

comillaSencilla = '\''

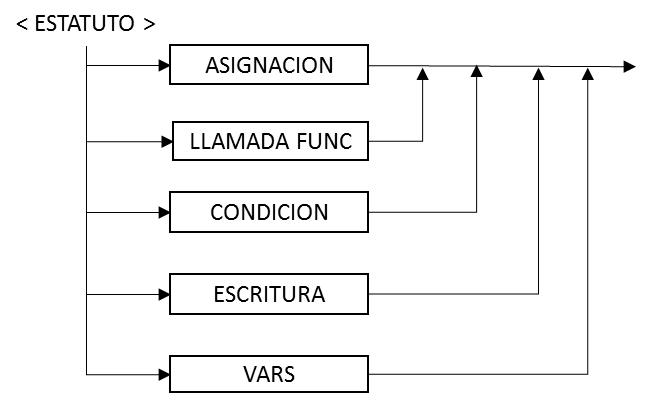
espacioEnBlanco = ' '

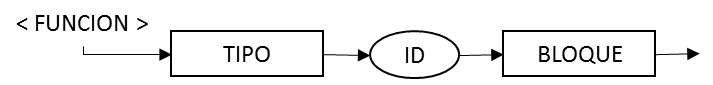
* ID : l (l|d)\*
* CTE INT : d (d)\*
* CTE FLOAT : d (d)\* "." d (d)\*
* CTE STRING : comillaSencilla (l | d | espacioEnBlanco)\* comillaSencilla

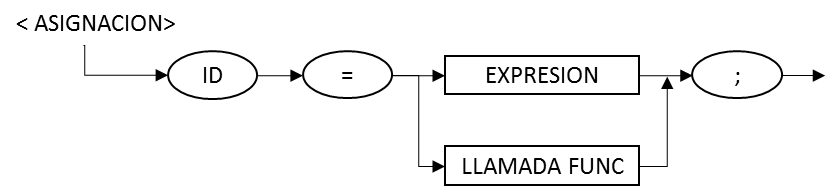
## DIAGRAMAS DE SINTAXIS



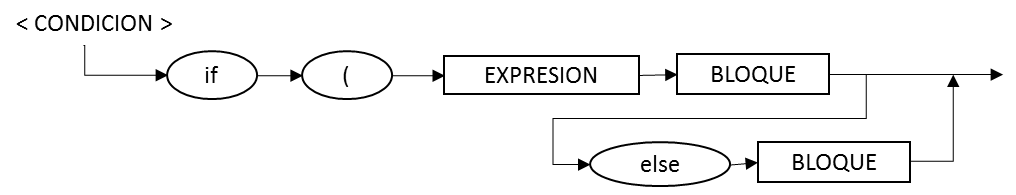


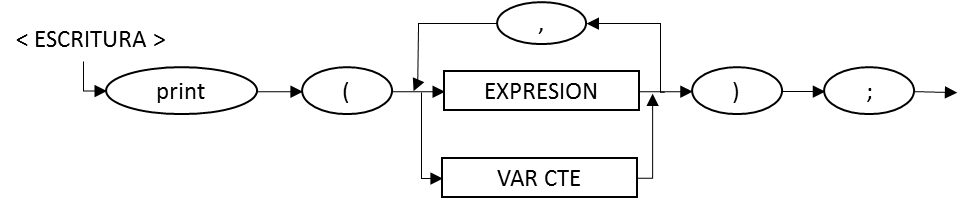


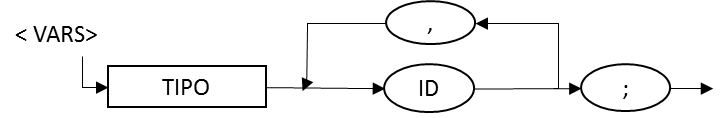


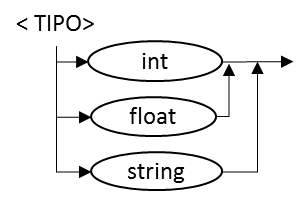


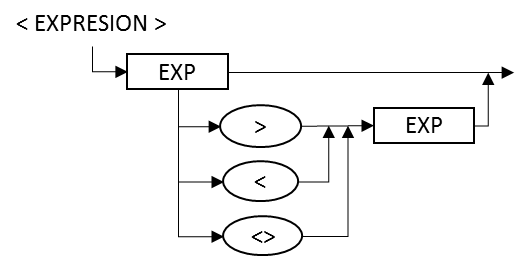


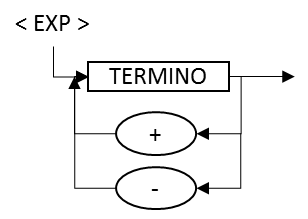


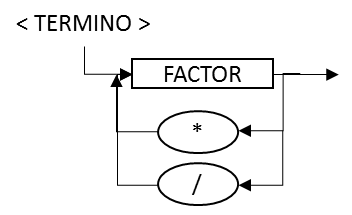


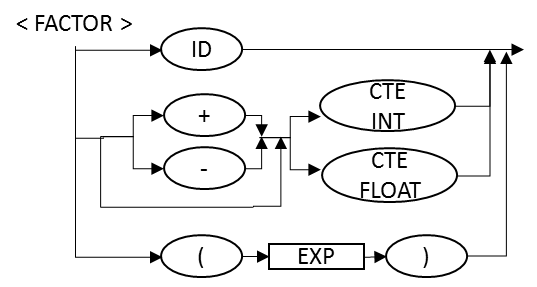


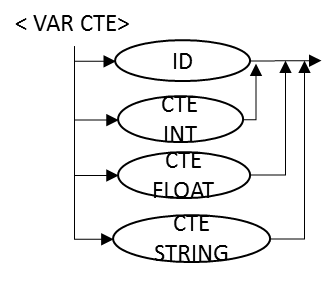












## PRINCIPALES CARACTERÍSTICAS SEMÁNTICAS

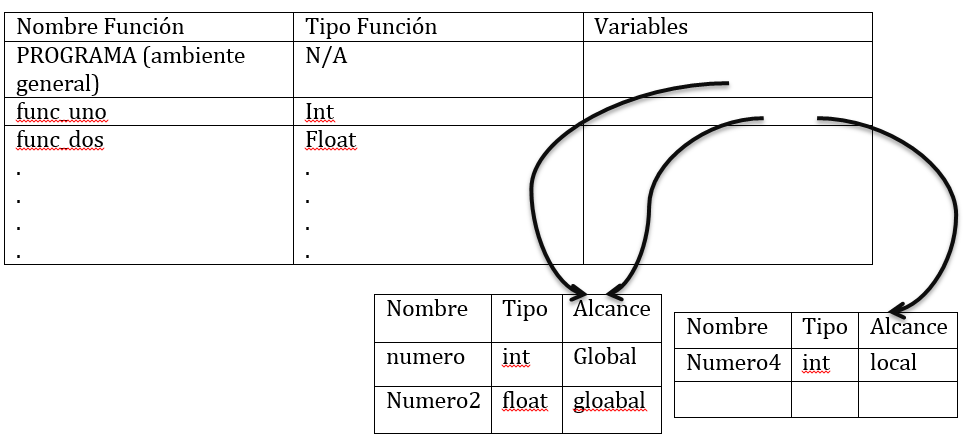
En relación a la cuestión semántica de “Funpiling”, hicimos una subdivisión entre los tres aspectos más importantes de la composición de un lenguaje: a nivel de variables, de expresiones y estatutos.

VARIABLES

En relación con las variables dentro de nuestro lenguaje de programación, proveeremos la capacidad de definición de éstas a nivel local y global. En otras palabras, el alcance de las variables podrá tener un impacto sobre todo el curso de una ejecución de código o serán específicas a un bloque anidado, haciéndolas locales. Para esto, se realizará la construcción de una tabla de referencia a procedimientos y, anidada a cada una de estos procedimientos, ira otra tabla pero relacionada a variables.

Cabe resaltar que cada una de los procedimientos anidados tendrá una relación adicional en su especificación en la tabla al aspecto global de variables. Un código ejecutable en nuestro lenguaje estará formado por mínimo un procedimiento (main). A este(os) se le podrán adjudicar variables del ambiente general.

Ej.



### EXPRESIONES

En cuanto a expresiones se refiere, aquí se lidia con la combinación de tipos de datos y la forma en que se realizan operaciones con dichos datos.

Respecto a las operaciones en nuestro lenguaje, cualquier operación obedecerá un orden de “evaluación” de izquierda a derecha (a menos de que sea una asignación), asignando las siguientes prioridades:

Operación aritmética

|  |  |
| --- | --- |
| Símbolo | Prioridad |
| ( ) | 1 |
| + ó - | 3 |
| \* ó / | 2 |

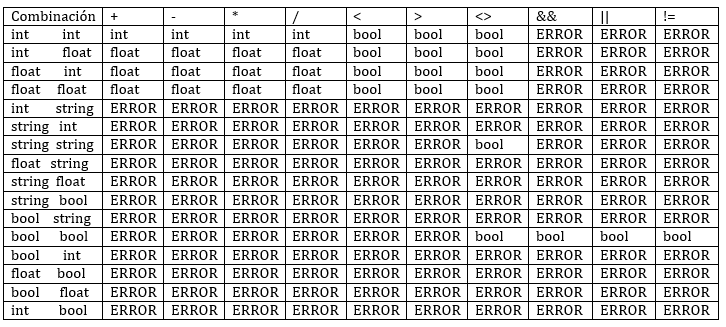
Operación booleana

|  |  |
| --- | --- |
| Símbolo | Prioridad |
| ( ) | 1 |
| <,>,<> | 2 |

|  |  |
| --- | --- |
| Tipo de Asociación | Injerencia |
| Izquierda | Para la evaluación de operaciones aritméticas y booleanas. |
| Derecha | Para la asignación de variables. |

### COMBINACIÓN DE TIPOS

Como mencionado anteriormente, tendremos 4 tipos de datos: strings, enteros, flotantes y booleanos. Es decir, de posibles combinaciones, 2^4 (16) habría. Cabe resaltar que no todas serán “legales”.



## FUNCIONES ESPECIALES DEL LENGUAJE

Por el momento, el equipo sólo visualiza como una función especial la de impresión. Las demás funciones especiales del lenguaje se obtendrán a través del uso de la interfaz.

## TIPOS DE DATOS

Los tipos de datos manejados por “Funpiling” son los siguientes:

|  |
| --- |
| int |
| float |
| string |
| bool |

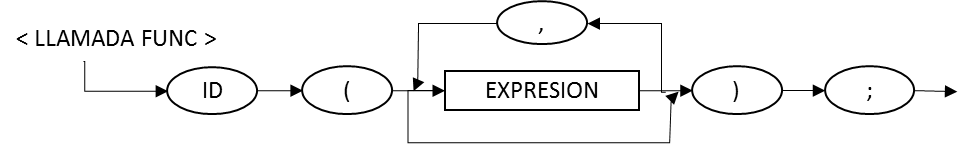
# PLATAFORMA DE DESARROLLO

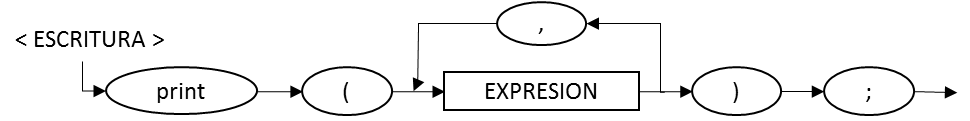
Se hará uso de Python v3.3, con el “framework” de PLY v3.4 Así mismo se emplearán las librerías de OpenGL v3.0.2 para Python. El ambiente de desarrollo seleccionado para nuestro proyecto será IDLE v3.3.3 para Mac.

# ANEXO 1 - PROTOTIPO

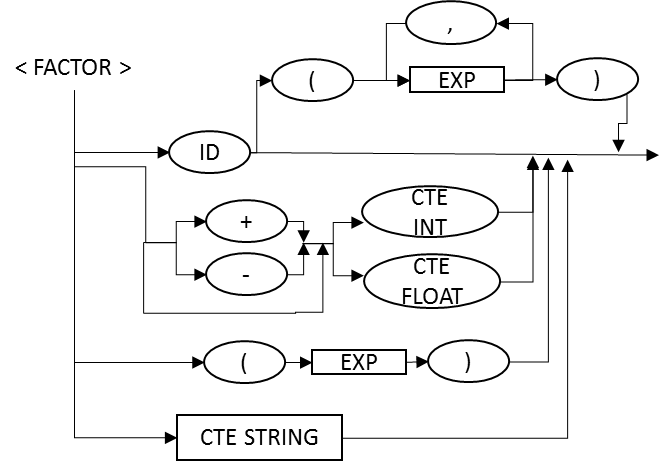
# ANEXO 2 – CORRECCIONES

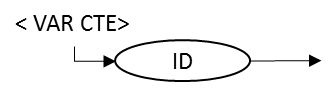


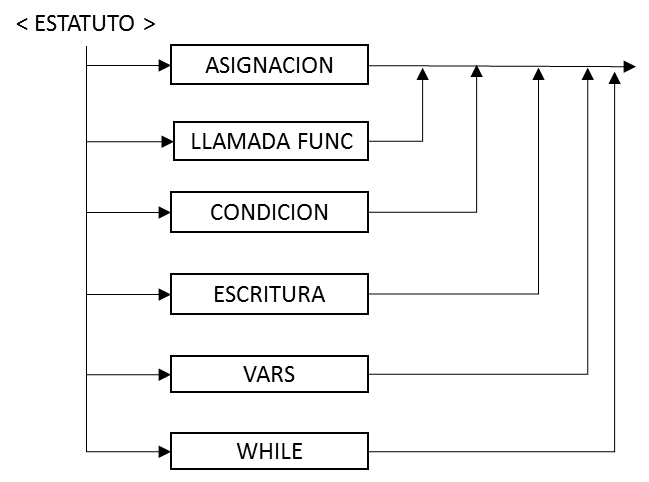












# BIBLIOGRAFÍA

* "LEGO®." *LEGO.com Mindstorms*. N.p., n.d. Web. 25 de febrero de 2014. <http://www.lego.com/en-us/mindstorms/?domainredir=mindstorms.lego.com>.
* "PLY (Python Lex-Yacc)." *PLY (Python Lex-Yacc)*. N.p., n.d. Web. 25 de febrero de 2014. <http://www.dabeaz.com/ply/ply.html>.
* "PyOpenGL 3.x." *PyOpenGL*. N.p., n.d. Web. 25 de febrero de 2014. <http://pyopengl.sourceforge.net/>.
* Quiroga, Elda y Roffe, Norma. *Traductores*. Web. 25 de febrero de 2014. <http://itc.mty.itesm.mx/equiroga/Material\_Traductores.pdf>
* Scott, Michael Lee. "Capítulo 2." *Programming Language Pragmatics*. San Francisco: Morgan Kaufmann, 2000. PDF.