

**Documentación Final**

**Proyecto de diseño de compiladores**

**TC3040**

**Snappy Code**

**(PLY)**

**Autores**:

**Juan Paulo Lara Rodríguez**

**A00999823**

**Manuel Calzado Maycotte**

**A00811102**

**30/Abril/2015**

INDICE

[Visión 5](#_Toc418559533)

[Objetivo 5](#_Toc418559534)

[Alcance 5](#_Toc418559535)

[Objetivo del lenguaje 5](#_Toc418559536)

[Análisis de Requerimientos 5](#_Toc418559537)

[Descripción de casos de uso 6](#_Toc418559538)

[Descripción de los principales Test Cases 8](#_Toc418559539)

[Descripción del proceso 8](#_Toc418559540)

[Descripción General 8](#_Toc418559541)

[Bitácoras 8](#_Toc418559542)

[Aprendizajes 13](#_Toc418559543)

[Manuel Calzado Maycotte 13](#_Toc418559544)

[Juan Paulo Lara Rodríguez 14](#_Toc418559545)

[Descripción del Lenguaje SnappyCode 14](#_Toc418559546)

[Descripción de los errores de compilación o ejecución 14](#_Toc418559547)

[Descripción del Compilador 15](#_Toc418559548)

[Descripción del hardware en el cual correrá́ el prototipo 15](#_Toc418559549)

[Descripción del lenguaje utilizado 16](#_Toc418559550)

[Python 16](#_Toc418559551)

[Descripción de utilerías especiales usadas en el desarrollo del proyecto 16](#_Toc418559552)

[PLY 16](#_Toc418559553)

[Descripción del Análisis de Léxico 16](#_Toc418559554)

[Patrones de Construcción de los elementos principales. 16](#_Toc418559555)

[Descripción del Análisis de Sintaxis 18](#_Toc418559556)

[Gramática Formal empleada. 18](#_Toc418559557)

[Descripción de Generación de Código Intermedio y Análisis Semántico 21](#_Toc418559558)

[Acciones 37](#_Toc418559559)

[Tabla de consideraciones semánticas. 37](#_Toc418559560)

[Descripción detallada del proceso de Administración de Memoria usado en la compilación. 38](#_Toc418559561)

[Descripción de la Máquina Virtual 39](#_Toc418559562)

[Pruebas del Funcionamiento del Lenguaje 42](#_Toc418559563)

[Cálculo de un Factorial recursivo 42](#_Toc418559564)

[Código spy: 42](#_Toc418559565)

[Cuádruplos: 42](#_Toc418559566)

[Resultado: 43](#_Toc418559567)

[El cálculo de un Factorial cíclico 43](#_Toc418559568)

[Código spy: 43](#_Toc418559569)

[Cuádruplos: 43](#_Toc418559570)

[Resultado: 43](#_Toc418559571)

[Cálculo de la serie de Fibonacci 44](#_Toc418559572)

[Código spy: 44](#_Toc418559573)

[Cuádruplos 44](#_Toc418559574)

[Resultado: 45](#_Toc418559575)

[Buscar un elemento en un arreglo 46](#_Toc418559576)

[Código spy 46](#_Toc418559577)

[Cuádruplos 46](#_Toc418559578)

[Resultado: 46](#_Toc418559579)

[Ordenar un arreglo 47](#_Toc418559580)

[Código spy: 47](#_Toc418559581)

[Cuádruplos: 47](#_Toc418559582)

[Resultado: 48](#_Toc418559583)

[Una multiplicación de arreglos 49](#_Toc418559584)

[Código spy: 49](#_Toc418559585)

[Cuádruplos: 49](#_Toc418559586)

[Resultado: 50](#_Toc418559587)

[Dibujar figuras geométricas y fractal 50](#_Toc418559588)

[Código spy: 50](#_Toc418559589)

[Cuádruplos: 51](#_Toc418559590)

[Resultado: 54](#_Toc418559591)

[ANEXO 1: Manual de Usuario 57](#_Toc418559592)

# Visión

Convertirse en una herramienta innovadora que permita a las personas, desde una edad temprana, aprender los fundamentos de programación de una manera fácil y atractiva.

# Objetivo

El objetivo del lenguaje “Snappy Code” es facilitar el aprendizaje de jóvenes en el área específica de la programación a través del uso de un lenguaje sencillo y muy detallado, demostrando a los alumnos las funcionalidades básicas de un lenguaje de programación.

# Alcance

Lenguaje de programación de alto nivel con soporte de estructuras de control de tipo condicionales y cíclicas, así como funcionalidades recursivas. Permite generar estatutos con resultados en consola o en una ventana de output gráfico.

# Objetivo del lenguaje

## Análisis de Requerimientos

Esta parte incluye la especificación de los requerimientos fundamentales para las acciones del lenguaje SnappyCode.

|  |  |
| --- | --- |
| **#** | **Especificación de requerimientos** |
| 1 | Output gráfico |
| 2 | Estatutos de asignación |
| 3 | Condiciones |
| 4 | Ciclos |
| 5 | Lectura de consola |
| 6 | Escritura en consola |
| 7 | Expresiones matemáticas |
| 8 | Funciones |
| 9 | Variables locales |
| 10 | Variables Globales |
| 11 | Arreglos |

## Descripción de casos de uso

|  |  |
| --- | --- |
| **ID** | **SPR1** |
| Título | Output gráfico |
| Descripción | Se crea una ventana de output gráfico en la cual el usuario realiza movimientos sobre un vector para dibujar |

|  |  |
| --- | --- |
| **ID** | **SPR2** |
| Título | Estatutos de asignación |
| Descripción | El usuario tiene la capacidad de asignar valores a las variables |

|  |  |
| --- | --- |
| **ID** | **SPR3** |
| Título | Condiciones |
| Descripción | El usuario puede realizar condiciones sobre variables o valores que desencadenan acciones. |

|  |  |
| --- | --- |
| **ID** | **SPR4** |
| Título | Ciclos |
| Descripción | El usuario puede ejecutar ciertas acciones en base a condiciones las cuales se repiten mientras se cumpla la acción |

|  |  |
| --- | --- |
| **ID** | **SPR5** |
| Título | Lectura de consola |
| Descripción | El usuario puede almacenar datos en una variable a partir de datos introducidos en consola |

|  |  |
| --- | --- |
| **ID** | **SPR6** |
| Título | Escritura en consola |
| Descripción | Se pueden imprimir datos en la consola a partir de constantes o expresiones |

|  |  |
| --- | --- |
| **ID** | **SPR7** |
| Título | Expresiones matemáticas |
| Descripción | El usuario puede realizar operaciones aritméticas y de comprobación sobre constantes u otras expresiones |

|  |  |
| --- | --- |
| **ID** | **SPR8** |
| Título | Funciones |
| Descripción | El usuario puede crear funciones las cuales pueden ser utilizadas más adelante, en estas funciones se pueden declarar distintas acciones |

|  |  |
| --- | --- |
| **ID** | **SPR9** |
| Título | Variables locales |
| Descripción | El usuario puede declarar variables dentro de las funciones las cuales solo pueden ser utilizadas dentro de estas. |

|  |  |
| --- | --- |
| **ID** | **SPR10** |
| Título | Variables Globales |
| Descripción | El usuario puede declarar variables al inicio del programa las cuales pueden ser utilizadas para cualquier función |

|  |  |
| --- | --- |
| **ID** | **SPR11** |
| Título | Arreglos |
| Descripción | El usuario puede crear una estructura de datos del tipo arreglo para almacenamiento y acceso a datos. |

## Descripción de los principales Test Cases

|  |  |
| --- | --- |
| **Caso de prueba** | **Descripción** |
| Cálculo de un Factorial recursivo | En este caso de prueba se va a probar la funcionalidad de la recursividad a través del cálculo de un número factorial. |
| El cálculo de un Factorial cíclico | En este caso de prueba se va a probar la funcionalidad de la de los ciclos a través del cálculo de un número factorial. |
| Cálculo de la serie de Fibonacci | Para este caso de prueba se pretende probar varias funcionalidades al hacer el cálculo de una serie de fibonacci. |
| Buscar un elemento en un arreglo | En esta prueba se realizará la búsqueda de un valor entre todos las casillas de un arreglo. |
| Ordenar un arreglo | En este caso de prueba se dará un arreglo desordenado y se debe mostrar el mismo arreglo pero ordenado los valores de menor a mayor . |
| Una multiplicación de arreglos | En esta prueba se realiza una suma de la multiplicación de cada uno de los valores de un arreglo. |
| Dibujar figuras geométricas y fractal | En este caso de prueba se pretenden probar todas las funciones de output gráfico mediante el trazado de varias imágenes de complejidades distintas. |

# Descripción del proceso

## Descripción General

Se realizó un amplio análisis de requerimientos previo al diseño, posteriormente se planteó la estructura general del lenguaje y se desarrolló bajo un esquema incremental en el cual se generaron módulos funcionales específicos para satisfacer requerimientos.

## Bitácoras

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Proyecto | SnappyCode | | Fecha de la Reunión | 4 de Marzo del 2015. | | Lugar | Biblioteca del ITESM, 4to Piso. | | Horario | 18:00 | | | | |
| Asistentes | Manuel Calzado Maycotte.  Juan Paulo Lara Rodríguez. | | |
|  | | | |
| Temas generales | | | |
| Análisis de Léxico y Sintaxis. | | | |
|  | | | |
| Acuerdos | | | |
| Se actualizarán las reglas de léxico y sintaxis para cumplir con los requerimientos. | | | |
|  | | | |
| Lista de Compromisos | | Responsable | FECHA DE terminación |
| Actualizarán reglas léxicas. | | Juan Paulo | 6 de Marzo del 2015. |
| Actualizarán reglas sintácticas. | | Manuel | 6 de Marzo del 2015. |
|  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Proyecto | SnappyCode | | Fecha de la Reunión | 9 de Marzo del 2015. | | Lugar | Biblioteca del ITESM, 4to Piso. | | Horario | 18:00 | | | | |
| Asistentes | Manuel Calzado Maycotte.  Juan Paulo Lara Rodríguez. | | |
|  | | | |
| Temas generales | | | |
| Tablas de Variables y de Procedimientos. | | | |
|  | | | |
| Acuerdos | | | |
| Se generara una tabla de variables individual para cada procedimiento. | | | |
|  | | | |
| Lista de Compromisos | | Responsable | FECHA DE terminación |
| Generar tabla de procedimientos. | | Juan Paulo | 13 de Marzo del 2015. |
| Generar tabla de variables individual para cada procedimiento. | | Manuel | 13 de Marzo del 2015. |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Proyecto | SnappyCode | | Fecha de la Reunión | 16 de Marzo del 2015. | | Lugar | Biblioteca del ITESM, 4to Piso. | | Horario | 18:00 | | | | |
| Asistentes | Manuel Calzado Maycotte.  Juan Paulo Lara Rodríguez. | | |
|  | | | |
| Temas generales | | | |
| Semántica de Expresiones. | | | |
|  | | | |
| Acuerdos | | | |
| Se generará el cubo semántico en conjunto, y el resto de los componentes para la entrega de manera individual. | | | |
|  | | | |
| Lista de Compromisos | | Responsable | FECHA DE terminación |
| Generacion de Codigo de Expresiones Aritméticas. | | Juan Paulo | 20 de Marzo del 2015. |
| Generar de Estatutos secuenciales: Asignación, lectura y escritura. | | Manuel | 20 de Marzo del 2015. |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Proyecto | SnappyCode | | Fecha de la Reunión | 23 de Marzo del 2015. | | Lugar | Biblioteca del ITESM, 4to Piso. | | Horario | 18:00 | | | | |
| Asistentes | Manuel Calzado Maycotte.  Juan Paulo Lara Rodríguez. | | |
|  | | | |
| Temas generales | | | |
| Estatutos Condicionales. | | | |
|  | | | |
| Acuerdos | | | |
| Únicamente se generará un estatuto de control de decisión y uno de cíclico. | | | |
|  | | | |
| Lista de Compromisos | | Responsable | FECHA DE terminación |
| Generar estatutos correspondientes a una estructura de decisión | | Juan Paulo | 20 de Marzo del 2015. |
| Generar estatutos correspondientes a una estructura cíclica. | | Manuel | 20 de Marzo del 2015. |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Proyecto | SnappyCode | | Fecha de la Reunión | 6 de Abril del 2015. | | Lugar | Biblioteca del ITESM, 4to Piso. | | Horario | 18:00 | | | | |
| Asistentes | Manuel Calzado Maycotte.  Juan Paulo Lara Rodríguez. | | |
|  | | | |
| Temas generales | | | |
| Generación de código para funciones. | | | |
|  | | | |
| Acuerdos | | | |
| Debido a la complejidad del tema se programara esta entrega utilizando la técnica de pair programming. | | | |
|  | | | |
| Lista de Compromisos | | Responsable | FECHA DE terminación |
| Generación de cuadruplos para llamada a funciones. | | Juan Paulo y Manuel | 10 de Abril del 2015. |
|  | |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Proyecto | SnappyCode | | Fecha de la Reunión | 13 de Abril del 2015. | | Lugar | Biblioteca del ITESM, 4to Piso. | | Horario | 18:00 | | | | |
| Asistentes | Manuel Calzado Maycotte.  Juan Paulo Lara Rodriguez. | | |
|  | | | |
| Temas generales | | | |
| Generación de código para funciones. | | | |
|  | | | |
| Acuerdos | | | |
| Se generará únicamente la parte de máquina virtual para expresiones aritméticas. | | | |
|  | | | |
| Lista de Compromisos | | Responsable | FECHA DE terminación |
| Generación de estatutos de la máquina virtual para interpretación de expresiones aritméticas. | | Juan Paulo y Manuel | 17 de Abril del 2015. |
|  | |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Proyecto | SnappyCode | | Fecha de la Reunión | 13 de Abril del 2015. | | Lugar | Biblioteca del ITESM, 4to Piso. | | Horario | 18:00 | | | | |
| Asistentes | Manuel Calzado Maycotte.  Juan Paulo Lara Rodríguez. | | |
|  | | | |
| Temas generales | | | |
| Generación de código para funciones. | | | |
|  | | | |
| Acuerdos | | | |
| Se generará únicamente la parte de máquina virtual para expresiones aritméticas. | | | |
|  | | | |
| Lista de Compromisos | | Responsable | FECHA DE terminación |
| Generación de estatutos de la máquina virtual para interpretación de expresiones aritméticas. | | Juan Paulo y Manuel | 17 de Abril del 2015. |
|  | |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Proyecto | SnappyCode | | Fecha de la Reunión | 20 de Abril del 2015. | | Lugar | Biblioteca del ITESM, 4to Piso. | | Horario | 18:00 | | | | |
| Asistentes | Manuel Calzado Maycotte.  Juan Paulo Lara Rodríguez. | | |
|  | | | |
| Temas generales | | | |
| Manejo de Arreglos y Estatutos Condicionales. | | | |
|  | | | |
| Acuerdos | | | |
| Se terminará el proyecto (sin documentación) totalmente para esta entrega. | | | |
|  | | | |
| Lista de Compromisos | | Responsable | FECHA DE terminación |
| Generar estatutos de interpretación de cuádruplos para llamada y asignación de arreglos. | | Juan Paulo | 24 de Abril del 2015. |
| Generar estatutos de interpretación de cuádruplos de estructuras de control. | | Manuel | 24 de Abril del 2015. |

# Aprendizajes

## Manuel Calzado Maycotte

La realización de este proyecto sin duda alguna fue una experiencia enriquecedora, la cual me dejo muchos nuevos conocimientos, en especial en relación al manejo de memoria en los lenguajes de programación, también en la importancia de la sintaxis y el léxico. Me ayudo a tener un mejor entendimiento de cómo es el funcionamiento de los estatutos de control de manera interna. También aprendí acerca de todo el proceso que se requiere para que el código que escribimos en un lenguaje llegue a ser ejecutado y mi último aprendizaje fue el conocimiento de un nuevo lenguaje de programación, python.

## Juan Paulo Lara Rodríguez

Durante el desarrollo de nuestro propio lenguaje mi principal aprendizaje fue saber la manera en que están construidos los lenguajes de programación que utilizamos día a día, esto nos ayuda a poder tener un entendimiento más profundo de cómo ellos. Aprendí también cómo funciona el proceso completo de compilación desde la generación de código intermedio que son los cuádruplos los cuales son muy parecidos a un lenguaje ensamblador hasta la interpretación de estos en la máquina virtual.

Por otro lado un muy gran aprendizaje fue el uso de un lenguaje nuevo que en este caso fue python ya que nunca antes lo había utilizado.

# Descripción del Lenguaje SnappyCode

Una de las características más importantes del lenguaje SnappyCode es la escritura se hace español y de manera muy detallada facilitando así el entendimiento de que es lo que realiza cada una de las instrucciones.

Otra funcionalidad muy importante es la de mostrar al usuario gráficamente las instrucciones que va realizando para así generar un mejor entendimiento de lo que realiza cada acción.

Utiliza una delimitación de bloques definidos por las palabras inicio y fin seguido del tipo de acción que se está ejecutando para cada una de las acciones que involucran más acciones.

## Descripción de los errores de compilación o ejecución

* Error de sintaxis
* Error Semántico: Valor de retorno incompatible con tipo de función
* Error Semántico: Variable no encontrada para asignación
* Error Semántico: Variable incompatible con valor a asignar
* Error Semántico: valores incompatibles en la comparación
* Error Semántico: valores incompatibles en suma
* Error Semántico: Variable no encontrada
* Parámetros incompatibles en llamada a función
* Cantidad de parámetros en llamada de la función 'es incorrecta
* El procedimiento no ha sido declarado
* Error Semántico: Variable de tipo vector no encontrada
* Error Semántico: Variable de tipo vector no encontrada
* Error Semántico: Variable incompatible con valor a asignar
* Error Semántico: Variable incompatible con valor a asignar
* Error Semántico: Variable incompatible con valor a asignar
* Índice fuera del rango del arreglo
* Error: No puede realizar divisiones entre 0.
* Error Semántico: Procedimiento ya fue declarado
* Error Semántico: Variable ya fue declarada
* Error Semántico: Variable ya fue declarada

# Descripción del Compilador

## Descripción del hardware en el cual correrá́ el prototipo

|  |  |
| --- | --- |
| Marca | Apple |
| Serie | C17M4GLYF5V7 |
| Número de modelo del producto | MacBook Air (13-inch, Mid 2013) |
| Dimensión de la pantalla | 13” |
| Resolución de pantalla | 1440 x 900 |
| Fabricante del procesador | Intel |
| Procesador | 1.3 GHz Intel Core i5 |
| Capacidad de la memoria RAM | 4 GB 1600 MHz DDR3 |
| Capacidad del disco duro | 120 GB SSD |
| Sistema operativo | OS X Yosemite 10.10.3 (14D136) |

|  |  |
| --- | --- |
| Marca | Hewlett-Packard |
| Serie | Hp Pavillion Sleekbook |
| Número de modelo del producto | 15-b119wm |
| Dimensión de la pantalla | 15.6-inch diagonal HD BrightView LED-backlit display |
| Resolución de pantalla | 1366x768 px |
| Fabricante del procesador | AMD |
| Procesador | 1.6GHz DualCore |
| Capacidad de la memoria RAM | 4 GB DDR3 |
| Capacidad del disco duro | 500 GB |
| Sistema operativo | Windows 8.1 |

## Descripción del lenguaje utilizado

### Python

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible.

Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico y es multiplataforma.

## Descripción de utilerías especiales usadas en el desarrollo del proyecto

### PLY

PLY o python-ply (cómo se llama su paquete en Debian) es una implementación de las herramientas lex y yacc para análisis léxico y sintáctico. Está enteramente escrito en Python y su primera versión fue desarrollado por David Beazley en el año 2001 para ser usado en un curso de Introducción a los Compiladores.

# Descripción del Análisis de Léxico

## Patrones de Construcción de los elementos principales.

t\_PUNTOCOMA = r';'

t\_PARENTIZQ = r'\('

t\_PARENTDER = r'\)'

t\_CORCHETEIZQ = r'\['

t\_CORCHETEDER = r'\]'

t\_IGUAL = r'='

t\_IGUALQUE = r'=='

t\_MAS = r'\+'

t\_MENOS = r'-'

t\_DIV = r'\/'

t\_MULT = r'\\*'

t\_MENORQUE = r'<'

t\_MAYORQUE = r'>'

t\_DIFERENTEQUE = r'<>'

t\_MAYORIGUAL = r'>='

t\_MENORIGUAL = r'<='

t\_COMA = r','

t\_TRUE = r'[1]'

t\_FALSE = r'[0]'

t\_ignore = " \t"

def t\_ID(t):

r'[A-Za-z]+[A-Za-z0-9]\*'

t.type = reserved.get(t.value,"ID")

return t

def t\_newline(t):

r'\n+'

t.lexer.lineno += len(t.value)

def t\_CTEFLOTANTE(t):

r'-?\d+\.\d\*(e-?\d+)?'

t.value = float(t.value)

return t

def t\_CTEENTERO(t):

r'\d+'

t.value = int(t.value)

return t

def t\_CTETEXTO(t):

r'\"[^\"]\*\"'

t.value = t.value

return t

def t\_error(t):

print("Caracter no valido'%s'" % t.value[0])

t.lexer.skip(1)

o Enumeración de los "tokens" del lenguaje y su código asociado. INICIOPROGRAMA

FINPROGRAMA

INICIOFUNCION

FINFUNCION

REGRESA

PARAMETROS

CREAR

BORRAR

ID

TRUE

FALSE

INICIOPRINCIPAL

FINPRINCIPAL

MOVER

GIRARDERECHA

GIRARIZQUIERDA

SI

FINS

ENTONCES

SINO

MIENTRAS

FINMIENTRAS

HACER

LISTA

IGUAL

IGUALQUE

MAYORQUE

MENORQUE

DIFERENTEQUE

PINTAR

DESPINTA

DECIRALUSUARIO

PEDIRALUSUARIO

INICIOBLOQUE

FINBLOQUE

MAS

MENOS

COMA

MULT

DIV

PUNTOCOMA

PARENTIZD

PARENTDER

ENTERO

FLOTANTE

TEXTO

CTEENTERO

CTEFLOTANTE

CTETEXTO

MAYORIGUAL

MENORIGUAL

BOOLEANO

CORCHETEIZQ

CORCHETEDER

# Descripción del Análisis de Sintaxis

## Gramática Formal empleada.

program ::= INICIOPROGRAMA vars cuerpo FINPROGRAMA

vars ::= vars tipo\_var

tipo\_var ::= single

| array

single ::= CREAR tipo ID PUNTOCOMA

array ::= CREAR tipo ID CORCHETEIZQ CTEENTERO CORCHETEDER PUNTOCOMA

tipo ::= ENTERO

| FLOTANTE

| TEXTO

| BOOLEANO

vars ::= empty

empty ::=

cuerpo ::= cuerpo\_func principal

cuerpo\_func ::= cuerpo\_func funcion

| empty

funcion ::= iniciofunc param vars finfunc

iniciofunc ::= INICIOFUNCION tipo ID

param ::= PARAMETROS tipo ID param\_mult

param\_mult ::= COMA tipo ID param\_mult

param\_mult ::= empty

param ::= empty

finfunc ::= C REGRESA expresion FINFUNCION

C ::= C estatuto

| empty

estatuto ::= asignacion

| condicion

| ciclo

| io

| accion

| llamada\_sin\_ret

asignacion ::= ID asignacion\_arreglo IGUAL expresion PUNTOCOMA

asignacion\_arreglo ::= CORCHETEIZQ exp CORCHETEDER

asignacion\_arreglo ::= empty

expresion ::= comparacion AND push\_opt comparacion

| comparacion OR push\_opt comparacion

comparacion ::= exp MAYORQUE push\_opt exp

| exp MENORQUE push\_opt exp

| exp DIFERENTEQUE push\_opt exp

| exp IGUALQUE push\_opt exp

| exp MAYORIGUAL push\_opt exp

| exp MENORIGUAL push\_opt exp

exp ::= PARENTIZQ expresion PARENTDER

exp ::= exp MAS push\_opt exp

| exp MENOS push\_opt exp

| exp MULT push\_opt exp

| exp DIV push\_opt exp

push\_opt ::= empty

exp ::= MENOS exp prec UMINUS

exp ::= CTEENTERO

exp ::= CTEFLOTANTE

exp ::= TRUE

| FALSE

exp ::= CTETEXTO

exp ::= ID varfuncarr

varfuncarr ::= push\_var\_opd

| llamada

| arraycall

push\_var\_opd ::= empty

llamada ::= PARENTIZQ llamada\_param PARENTDER

llamada\_param ::= expresion llamada\_param\_mult

expresion ::= comparacion

comparacion ::= exp

llamada\_param\_mult ::= llamada\_param\_mult COMA expresion

llamada\_param\_mult ::= empty

llamada\_param ::= empty

arraycall ::= CORCHETEIZQ exp CORCHETEDER

condicion ::= SI expresion actSi1 ENTONCES bloque actSi2 condicion\_else actSi3 FINSI

actSi1 ::= empty

bloque ::= INICIOBLOQUE bloque\_estatuto\_mult FINBLOQUE

bloque\_estatuto\_mult ::= bloque\_estatuto\_mult estatuto

| empty

actSi2 ::= empty

condicion\_else ::= SINO bloque

| empty

actSi3 ::= empty

ciclo ::= MIENTRAS actCic1 expresion actCic2 HACER bloque FINMIENTRAS actCic3

actCic1 ::= empty

actCic2 ::= empty

actCic3 ::= empty

io ::= DECIRALUSUARIO exp PUNTOCOMA

io ::= PEDIRALUSUARIO PARENTIZQ ID asignacion\_arreglo PARENTDER PUNTOCOMA

accion ::= tipo\_accion PUNTOCOMA

tipo\_accion ::= objeto\_con\_exp

| objeto\_sin\_exp

objeto\_con\_exp ::= GIRARDERECHA exp

| GIRARIZQUIERDA exp

| MOVER exp

objeto\_sin\_exp ::= BORRAR

| PINTAR

| DESPINTAR

llamada\_sin\_ret ::= ID llamada PUNTOCOMA

principal ::= iniciomain vars C FINPRINCIPAL

iniciomain ::= INICIOPRINCIPAL

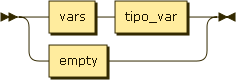
# Descripción de Generación de Código Intermedio y Análisis Semántico

**program:**

C:\Users\Maycot\Downloads\diagram (4)\diagram\program.png

|  |
| --- |
| Acción |
| Solicitar a la máquina virtual la interpretación del código intermedio generado. |

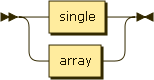
**vars:**



Referenciado por:

* [funcion](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#funcion)
* [principal](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#principal)
* [program](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#program)
* [vars](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#vars)

**tipo\_var:**



Referenciado por:

* [vars](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#vars)

**single:**

C:\Users\Maycot\Downloads\diagram (4)\diagram\single.png

Referenciado por:

* [tipo\_var](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#tipo_var)

|  |
| --- |
| Acción |
| Si el procedimiento actual es “global”, inserta la variable en la tabla de variables globales, sino inserta la variable en la tabla de variables locales del respectivo procedimiento |

**array:**

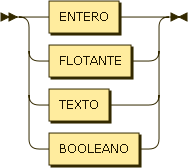
C:\Users\Maycot\Downloads\diagram (4)\diagram\array.png

Referenciado por:

* [tipo\_var](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#tipo_var)

|  |
| --- |
| Acción |
| Si el procedimiento actual es “global”, inserta la variable en la tabla de variables globales indicando su dimensión, sino inserta la variable en la tabla de variables locales del respectivo procedimiento indicando su dimensión. |

**tipo:**



Referenciado por:

* [array](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#array)
* [iniciofunc](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#iniciofunc)
* [param](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#param)
* [param\_mult](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#param_mult)
* [single](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#single)

**empty:**

C:\Users\Maycot\Downloads\diagram (4)\diagram\empty.png

Referenciado por:

* [C](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#C)
* [actCic1](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#actCic1)
* [actCic2](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#actCic2)
* [actCic3](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#actCic3)
* [actSi1](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#actSi1)
* [actSi2](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#actSi2)
* [actSi3](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#actSi3)
* [asignacion\_arreglo](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#asignacion_arreglo)
* [bloque\_estatuto\_mult](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#bloque_estatuto_mult)
* [condicion\_else](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#condicion_else)
* [cuerpo\_func](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#cuerpo_func)
* [llamada\_param](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#llamada_param)
* [llamada\_param\_mult](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#llamada_param_mult)
* [param](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#param)
* [param\_mult](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#param_mult)
* [push\_opt](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#push_opt)
* [push\_var\_opd](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#push_var_opd)
* [vars](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#vars)

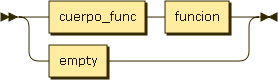
**cuerpo:**

C:\Users\Maycot\Downloads\diagram (4)\diagram\cuerpo.png

Referenciado por:

* [program](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#program)

**cuerpo\_func:**



Referenciado por:

* [cuerpo](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#cuerpo)
* [cuerpo\_func](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#cuerpo_func)

**funcion:**

C:\Users\Maycot\Downloads\diagram (4)\diagram\funcion.png

Referenciado por:

* [cuerpo\_func](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#cuerpo_func)

**iniciofunc:**

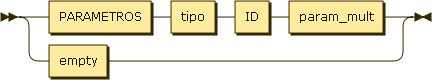
C:\Users\Maycot\Downloads\diagram (4)\diagram\iniciofunc.png

Referenciado por:

* [funcion](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#funcion)

|  |
| --- |
| Acción |
| Genera una variable temporal que será la variable de retorno de la función. Inserta el procedimiento en la tabla de procedimientos |

**param:**

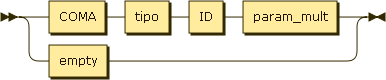


Referenciado por:

* [funcion](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#funcion)

|  |
| --- |
| Acción |
| En caso de recibir el token PARAMETROS ingresa la variable en tabla de parámetros del procedimiento actual. |

**param\_mult:**



Referenciado por:

* [param](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#param)
* [param\_mult](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#param_mult)

|  |
| --- |
| Acción |
| En caso de recibir el token COMA ingresa la variable en tabla de parámetros del procedimiento actual. |

**finfunc:**

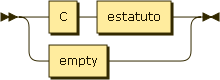
C:\Users\Maycot\Downloads\diagram (4)\diagram\finfunc.png

Referenciado por:

* [funcion](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#funcion)

|  |
| --- |
| Acción |
| Genera el cuádruplo con operador ENDPROC indicando el final de una función. Saca de la pila de operandos la última variable a través de un pop y asigna la dirección de dicha variable al valor de la variable de retorno de la función actual. |

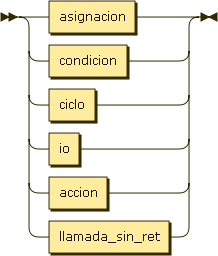
**C:**



Referenciado por:

* [C](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#C)
* [finfunc](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#finfunc)
* [principal](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#principal)

**estatuto:**



Referenciado por:

* [C](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#C)
* [bloque\_estatuto\_mult](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#bloque_estatuto_mult)

**asignacion:**

C:\Users\Maycot\Downloads\diagram (4)\diagram\asignacion.png

[asignacion](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#asignacion)

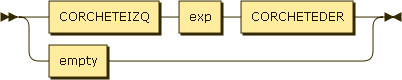
::= [ID](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#ID) [asignacion\_arreglo](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#asignacion_arreglo) [IGUAL](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#IGUAL) [expresion](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#expresion) [PUNTOCOMA](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#PUNTOCOMA)

Referenciado por:

* [estatuto](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#estatuto)

|  |
| --- |
| Acción |
| Busca el ID en la tablas de variables globales y en las tablas de variables y parámetros del procedimiento actual, obteniendo así la variable destino donde se guardara el valor de expresión. Saca de la pila de operandos la última variable a través de un pop y obtiene el valor a asignar a la variable destino. Genera el cuádruplo pertinente con operando “=”, indicando una asignación a una variable. En caso de que la regla “asignación arreglo” regrese algún valor realiza el mismo proceso solo que genera los cuádruplos con operador “VER”, “OFST”,“ARYAS” indicando una verificación, cálculo de la dirección del arreglo en base al índice y una asignación respectivamente. |

**asignacion\_arreglo:**

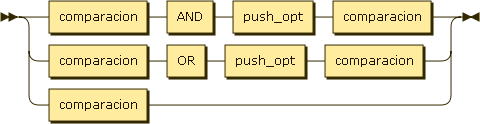


Referenciado por:

* [asignacion](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#asignacion)
* [io](file:///C:\Users\Maycot\Downloads\diagram%20(4)\index.html#io)

|  |
| --- |
| Acción |
| En caso de recibir el token CORCHETEIZQ lo retorna a la regla anterior. |

**expresion:**

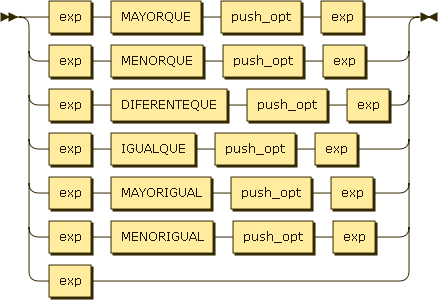


Referenciado por:

* [asignacion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#asignacion)
* [ciclo](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#ciclo)
* [condicion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#condicion)
* [exp](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#exp)
* [finfunc](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#finfunc)
* [llamada\_param](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#llamada_param)
* [llamada\_param\_mult](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#llamada_param_mult)

|  |
| --- |
| Acción |
| Realiza dos pops de la pila de operandos obteniendo el operando2 y el operando1 respectivamente. Realiza pop de la pila de operadores. Genera una variable temporal donde se almacenara el resultado de la operación. Se genera el cuádruplo pertinente con los siguientes posibles operadores: “||”, “&&”. En caso de error semántico, informa al usuario. |

**comparacion:**

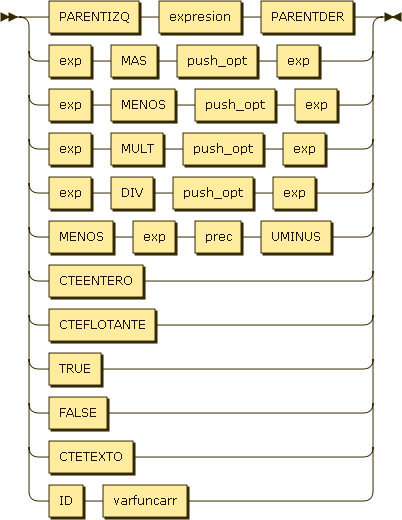


Referenciado por:

* [expresion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#expresion)

|  |
| --- |
| Acción |
| Realiza dos pops de la pila de operandos obteniendo el operando2 y el operando1 respectivamente. Realiza pop de la pila de operadores. Genera una variable temporal donde se almacenara el resultado de la operación. Se genera el cuádruplo pertinente con los siguientes posibles operadores: “>”, “<”,“!=”, “==”,“>=”, “<=”. En caso de error semántico, informa al usuario. |

**exp:**



Referenciado por:

* [arraycall](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#arraycall)
* [asignacion\_arreglo](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#asignacion_arreglo)
* [comparacion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#comparacion)
* [exp](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#exp)
* [io](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#io)
* objeto\_con\_exp

|  |
| --- |
| Acción |
| En caso de recibir regla “exp”. Realiza dos pops de la pila de operandos obteniendo el operando2 y el operando1 respectivamente. Realiza pop de la pila de operadores. Genera una variable temporal donde se almacenara el resultado de la operación. Se genera el cuádruplo pertinente con los siguientes posibles operadores: “+”, “-”,“\*”, “/”. En caso de error semántico, informa al usuario.  En caso de recibir regla “CTEENTERO”, “CTEFLOTANTE”, “CTETEXTO”, “CTEBOOLEANO”,”TRUE”,”FALSE”. Inserta el valor recibido de la regla en la tabla de constantes y lo inserta en la pila de operandos. |

**push\_opt:**

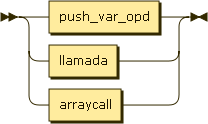
C:\Users\Maycot\Downloads\diagram (5)\diagram\push_opt.png

Referenciado por:

* [comparacion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#comparacion)
* [exp](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#exp)
* [expresion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#expresion)

|  |
| --- |
| Acción |
| Inserta el operador, contenido en el token anterior, en la pila de operadores. |

**varfuncarr:**



Referenciado por:

* [exp](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#exp)

**push\_var\_opd:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\push_var_opd.png

Referenciado por:

* [varfuncarr](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#varfuncarr)

|  |
| --- |
| Acción |
| Inserta el operando, contenido en el token anterior, en la pila de operandos. |

**llamada:**

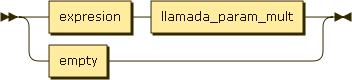
C:\Users\Maycot\Downloads\diagram (5)\diagram\llamada.png

Referenciado por:

* [llamada\_sin\_ret](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#llamada_sin_ret)
* [varfuncarr](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#varfuncarr)

|  |
| --- |
| Acción |
| Se obtiene el nombre del procedimiento del token anterior y se busca en la tabla de procedimientos, en caso de no existir se le informa al usuario del error semántico. Se generan cuádruplo de solicitud de memoria con operando “ERA”. Se generan cuádruplo de paso de control de ejecución con operando “GOSUB”. Se evalúa la semántica del llamado a la función comparando los parámetros de la llamada con los parámetros de la función. Se generan cuádruplos de envió de parámetros con operando “PARAM”. |

**llamada\_param:**

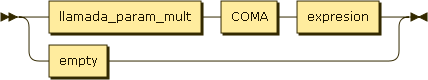


Referenciado por:

* [llamada](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#llamada)

|  |
| --- |
| Acción |
| Realiza pop a la pila de operandos y lo anexa a la pila temporal de parámetros. |

**llamada\_param\_mult:**



Referenciado por:

* [llamada\_param](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#llamada_param)
* [llamada\_param\_mult](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#llamada_param_mult)

|  |
| --- |
| Acción |
| En caso de recibir token “COMA”, realiza pop a la pila de operandos y lo anexa a la pila temporal de parámetros. |

**arraycall:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\arraycall.png

Referenciado por:

* [varfuncarr](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#varfuncarr)

|  |
| --- |
| Acción |
| Busca el ID contenido en el token anterior en la tablas de variables globales y en las tablas de variables y parámetros del procedimiento actual. Saca de la pila de operandos la última variable a través de un pop y obtiene el valor valor del índice del arreglo. Genera el cuádruplo pertinente con operando “VER”, indicando que es necesaria una verificación del índice contra las dimensiones del arreglo. Genera el cuádruplo pertinente con operando “ARYCA”, indicando que se está realizando una llamada a un arreglo. |

**condicion:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\condicion.png

Referenciado por:

* [estatuto](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#estatuto)

**actSi1:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\actSi1.png

Referenciado por:

* [condicion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#condicion)

|  |
| --- |
| Acción |
| Inserta en la pila de saltos el contador actual. Realiza un pop de la pila de operandos para obtener el operando 1. Genera el cuádruplo pertinente con operador “GOTOF” anexando el operando obtenido. |

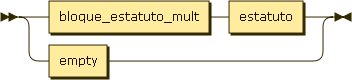
**bloque:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\bloque.png

Referenciado por:

* [ciclo](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#ciclo)
* [condicion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#condicion)
* [condicion\_else](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#condicion_else)

**bloque\_estatuto\_mult:**



Referenciado por:

* [bloque](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#bloque)
* [bloque\_estatuto\_mult](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#bloque_estatuto_mult)

**actSi2:**

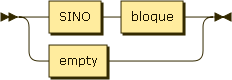
C:\Users\Maycot\Downloads\diagram (5)\diagram\actSi2.png

Referenciado por:

* [condicion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#condicion)

|  |
| --- |
| Acción |
| Genera un cuádruplo con operando “GOTO”. Realiza un pop de la pila de Saltos y navega hasta el cuádruplo indicado actualizando el operando 2 de dicho cuádruplo con el contador de cuádruplos actual. |

**condicion\_else:**



Referenciado por:

* [condicion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#condicion)

**actSi3:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\actSi3.png

Referenciado por:

* [condicion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#condicion)

|  |
| --- |
| Acción |
| Realiza un pop de la pila de Saltos y navega hasta el cuádruplo indicado actualizando el operando 2 de dicho cuádruplo con el contador de cuádruplos actual. |

**ciclo:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\ciclo.png

Referenciado por:

* [estatuto](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#estatuto)

**actCic1:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\actCic1.png

Referenciado por:

* [ciclo](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#ciclo)

|  |
| --- |
| Acción |
| Inserta en la pila de saltos el contador del cuádruplo actual. |

**actCic2:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\actCic2.png

Referenciado por:

* [ciclo](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#ciclo)

|  |
| --- |
| Acción |
| Realiza un pop de la pila de operandos para obtener el operando 1. Genera el cuádruplo pertinente con operador “GOTOF” anexando el operando obtenido. Inserta en la pila de saltos el contador actual menos uno. |

**actCic3:**

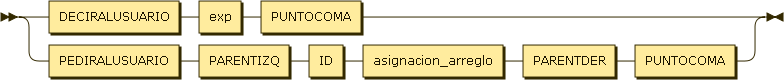
C:\Users\Maycot\Downloads\diagram (5)\diagram\actCic3.png

Referenciado por:

* [ciclo](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#ciclo)

|  |
| --- |
| Acción |
| Realiza un pop de la pila de operandos para obtener el operando 1. Realiza un pop de la pila de Saltos y navega hasta el cuádruplo indicado actualizando el operando 2 de dicho cuádruplo con el contador de cuádruplos actual más uno. Realiza un pop de la pila de operandos para obtener el operando 1. Genera el cuádruplo pertinente con operador “GOTO” anexando el operando obtenido. |

**io:**



Referenciado por:

* [estatuto](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#estatuto)

|  |
| --- |
| Acción |
| En caso de recibir el token “DECIRALUSUARIO”. Realiza un pop de la pila de operandos para obtener el operando 1. Genera el cuádruplo pertinente con operador “PRINT” anexando el operando obtenido.  En caso de recibir el token “PEDIRALUSUARIO”. Busca el ID en la tablas de variables globales y en las tablas de variables y parámetros del procedimiento actual, obteniendo así la variable destino donde se guardara el valor de la expresión ingresada por el usuario. Se genera una variable temporal. Se genera el cuádruplo pertinente con operador “INPUT” mandando con operador 1 la variable obtenida y como resultado el temporal generado. Si la regla “asignación\_arreglo” retorna valor, se generan los cuádruplos pertinentes con operadnos “VER”,”OFST” y “ARYAS”. |

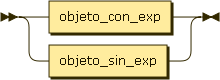
**accion:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\accion.png

Referenciado por:

* [estatuto](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#estatuto)

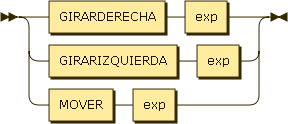
**tipo\_accion:**



Referenciado por:

* [accion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#accion)

**objeto\_con\_exp:**

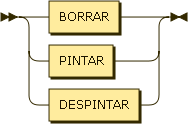


Referenciado por:

* [tipo\_accion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#tipo_accion)

|  |
| --- |
| Acción |
| Realiza un pop de la pila de operandos para obtener el operando 1. Se genera el cuádruplo pertinente con los siguientes operadores posibles: “GIRARDERECHA”,”GIRARIZQUIERDA”,”MOVER”. |

**objeto\_sin\_exp:**



Referenciado por:

* [tipo\_accion](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#tipo_accion)

|  |
| --- |
| Acción |
| Realiza un pop de la pila de operandos para obtener el operando 1. Se genera el cuádruplo pertinente con los siguientes operadores posibles: “BORRAR”,”PINTAR”,”DESPINTAR”. |

**llamada\_sin\_ret:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\llamada_sin_ret.png

Referenciado por:

* [estatuto](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#estatuto)

**principal:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\principal.png

Referenciado por:

* [cuerpo](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#cuerpo)

**iniciomain:**

C:\Users\Maycot\Downloads\diagram (5)\diagram\iniciomain.png

Referenciado por:

* [principal](file:///C:\Users\Maycot\Downloads\diagram%20(5)\index.html#principal)

# Acciones

## Tabla de consideraciones semánticas.

|  |  |  |  |
| --- | --- | --- | --- |
| entero | flotante | texto | booleano |
| "entero": {  "+":"entero",  "-":"entero",  "\*":"entero",  "/":"entero",  ">":"booleano",  ">=":"booleano",  "<=":"booleano",  "<":"booleano",  "=":"entero",  "==": "booleano",  "!=":"booleano",  "&&":"error",  "||":"error",  },  "flotante": {  "+":"flotante",  "-":"flotante",  "\*":"flotante",  "/":"flotante",  ">":"booleano",  ">=":"booleano",  "<=":"booleano",  "<":"booleano",  "=":"entero",  "==": "booleano",  "!=":"booleano",  "&&":"error",  "||":"error",  },  "texto": {  "+":"error",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"error",  "==": "error",  "!=":"error",  "&&":"error",  "||":"error",  },  "booleano": {  "+":"error",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"error",  "==": "error",  "!=":"error",  "&&":"error",  "||":"error" }, | "entero": {  "+":"flotante",  "-":"flotante",  "\*":"flotante",  "/":"flotante",  ">":"booleano",  ">=":"booleano",  "<=":"booleano",  "<":"booleano",  "=":"flotante",  "==": "booleano",  "!=":"booleano",  "&&":"error",  "||":"error",  },  "flotante": {  "+":"flotante",  "-":"flotante",  "\*":"flotante",  "/":"flotante",  ">":"booleano",  ">=":"booleano",  "<=":"booleano",  "<":"booleano",  "=":"flotante",  "==": "booleano",  "!=":"booleano",  "&&":"error",  "||":"error",  },  "texto": {  "+":"error",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"error",  "==": "error",  "!=":"error",  "&&":"error",  "||":"error",  },  "booleano": {  "+":"error",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"error",  "==": "error",  "!=":"error",  "&&":"error",  "||":"error",  }, | "entero": {  "+":"error",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"error",  "==": "error",  "!=":"error",  "&&":"error",  "||":"error",  },  "flotante": {  "+":"error",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"error",  "==": "error",  "!=":"error",  "&&":"error",  "||":"error",  },  "texto": {  "+":"texto",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"texto",  "==": "booleano",  "!=":"booleano",  "&&":"error",  "||":"error",  },  "booleano": {  "+":"error",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"error",  "==": "error",  "!=":"error",  "&&":"error",  "||":"error",  }, | "entero": {  "+":"error",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"error",  "==": "error",  "!=":"error",  "&&":"error",  "||":"error",  },  "flotante": {  "+":"error",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"error",  "==":"error",  "!=":"error",  "&&":"error",  "||":"error",  },  "texto": {  "+":"error",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"error",  "==":"error",  "!=":"error",  "&&":"error",  "||":"error",  },  "booleano": {  "+":"error",  "-":"error",  "\*":"error",  "/":"error",  ">":"error",  ">=":"error",  "<=":"error",  "<":"error",  "=":"booleano",  "==":"booleano",  "!=":"booleano",  "&&":"booleano",  "||":"booleano",  }, |

# Descripción detallada del proceso de Administración de Memoria usado en la compilación.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nodo de Tabla de Variables | | | | |
| Nombre | Valor | Tipo | Dirección | Dimensión |
| Texto | Dirección de memoria | Texto | Dirección de memoria | Numero |

|  |
| --- |
| Tabla de Variables |
| Nodo de Tabla de Variables |
| Nodo de Tabla de Variables |
| Nodo de Tabla de Variables |
| … |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nodo de Tabla de Procedimientos | | | | | |
| Nombre | Tipo | Variable de Retorno | Numero de Cuádruplo | Tabla de Variables | Tabla de Parámetros |
| Texto | Texto | Dirección de memoria a un temporal | Numero | Tabla de Variables | Tabla de Parametros |

|  |
| --- |
| Tabla de Procedimientos |
| Nodo de Tabla de Procedimientos |
| Nodo de Tabla de Procedimientos |
| Nodo de Tabla de Procedimientos |
| … |

|  |  |  |
| --- | --- | --- |
| Nodo de Tabla de Constantes | | |
| Valor | Tipo | Dirección |
| Puede ser un string, integer, boolean o float | Texto | Dirección de memoria |

|  |
| --- |
| Tabla de Constantes |
| Nodo de Tabla de Constantes |
| Nodo de Tabla de Constantes |
| Nodo de Tabla de Constantes |
| … |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nodo de Tabla de Cuadruplos | | | | |
| Numero de Cuádruplo | Operador | Operando1 | Operando2 | Resultado |
| Numero | Texto | Dirección de memoria | Dirección de memoria | Dirección de memoria |

|  |
| --- |
| Tabla de Variables |
| Nodo de Tabla de Variables |
| Nodo de Tabla de Variables |
| Nodo de Tabla de Variables |
| … |

# Descripción de la Máquina Virtual

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nodo de Tabla de Variables | | | | |
| Nombre | Valor | Tipo | Dirección | Dimensión |
| Texto | Dirección de memoria | Texto | Dirección de memoria | Numero |

|  |
| --- |
| Tabla de Variables |
| Nodo de Tabla de Variables |
| Nodo de Tabla de Variables |
| Nodo de Tabla de Variables |
| … |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nodo de Tabla de Procedimientos | | | | | |
| Nombre | Tipo | Variable de Retorno | Numero de Cuádruplo | Tabla de Variables | Tabla de Parámetros |
| Texto | Texto | Dirección de memoria a un temporal | Numero | Tabla de Variables | Tabla de Parametros |

|  |
| --- |
| Tabla de Procedimientos |
| Nodo de Tabla de Procedimientos |
| Nodo de Tabla de Procedimientos |
| Nodo de Tabla de Procedimientos |
| … |

|  |  |  |
| --- | --- | --- |
| Nodo de Tabla de Constantes | | |
| Valor | Tipo | Dirección |
| Puede ser un string, integer, boolean o float | Texto | Dirección de memoria |

|  |
| --- |
| Tabla de Constantes |
| Nodo de Tabla de Constantes |
| Nodo de Tabla de Constantes |
| Nodo de Tabla de Constantes |
| … |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nodo de Tabla de Cuadruplos | | | | |
| Numero de Cuádruplo | Operador | Operando1 | Operando2 | Resultado |
| Numero | Texto | Dirección de memoria | Dirección de memoria | Dirección de memoria |

|  |
| --- |
| Tabla de Variables |
| Nodo de Tabla de Variables |
| Nodo de Tabla de Variables |
| Nodo de Tabla de Variables |
| … |

|  |
| --- |
| Pila de Saltos de Ejecución |
| Numero de Cuádruplo |
| Numero de Cuádruplo |
| Numero de Cuádruplo |
| … |

|  |
| --- |
| Pila de Espacios de Memoria |
| Tabla de Variables |
| Tabla de Variables |
| Tabla de Variables |
| … |

|  |
| --- |
| Pila de Nombres de Espacios de Memoria |
| Nombre de procedimiento |
| Nombre de procedimiento |
| Nombre de procedimiento |
| … |

Se utilizaron los resultados obtenidos en compilación; tabla de cuádruplos, tabla de procedimientos con sus respectivas tablas de variables y parámetros, tabla de constantes y tabla de temporales.

Las direcciones de memoria se clasificaron de la siguiente manera:

· Globales:

o Entero de 0 a 999

o Flotante de 1000 a 1999

o Textol de 2000 a 2999

o Boolean de 3000 a 3999

· Locales

o Entero de 4000 a 4999

o Flotante de 5000 a 5999

o Textol de 6000 a 6999

o Boolean de 7000 a 7999

· Temporales

o Entero de 8000 a 8999

o Flotante de 9000 a 9999

o Texto de 10000 a 10999

o Boolean de 11000 a 11999

· Constantes

o Entero de 12000 a 12999

o Flotante de 13000 a 13999

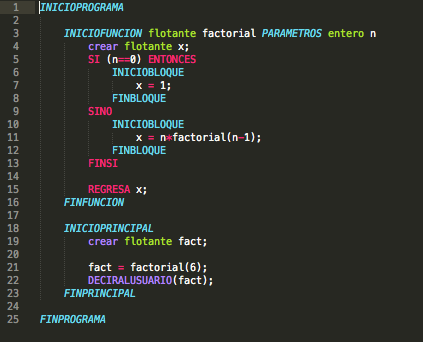
o Texto de 14000 a 14999

o Boolean de 15000 a 15999

# Pruebas del Funcionamiento del Lenguaje

## Cálculo de un Factorial recursivo

### Código spy:



### Cuádruplos:

0 | == | 4000 | 12000 | 11000

1 | GOTOF | 11000 | 4 | None

2 | = | 12001 | None | 5000

3 | GOTO | None | 10 | None

4 | - | 4000 | 12001 | 8000

5 | ERA | None | factorial | None

6 | GOSUB | None | factorial | None

7 | PARAM | 8000 | None | None

8 | \* | 4000 | 9000 | 9001

9 | = | 9001 | None | 5000

10 | ENDPROC | None | None | None

11 | ERA | None | factorial | None

12 | GOSUB | None | factorial | None

13 | PARAM | 12003 | None | None

14 | = | 9000 | None | 5001

15 | PRINT | 5001 | None | None

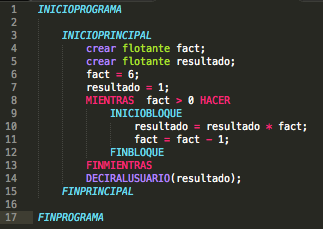
16 | ENDPROG | None | None | None

### Resultado:

Screen Shot 2015-05-04 at 10.07.37 PM.png

## El cálculo de un Factorial cíclico

### Código spy:



### Cuádruplos:

0 | = | 12000 | None | 5000

1 | = | 12001 | None | 5001

2 | > | 5000 | 12002 | 11000

3 | GOTOF | 11000 | 9 | None

4 | \* | 5001 | 5000 | 9000

5 | = | 9000 | None | 5001

6 | - | 5000 | 12001 | 9001

7 | = | 9001 | None | 5000

8 | GOTO | None | 2 | None

9 | PRINT | 5001 | None | None

10 | ENDPROG | None | None | None

### Resultado:

Screen Shot 2015-05-04 at 10.13.58 PM.png

## Cálculo de la serie de Fibonacci

### Código spy:



### Cuádruplos

0 | > | 4000 | 12000 | 11000

1 | GOTOF | 11000 | 15 | None

2 | - | 4000 | 12001 | 8001

3 | ERA | None | fiboAux | None

4 | GOSUB | None | fiboAux | None

5 | PARAM | 8001 | None | None

6 | = | 8000 | None | 4002

7 | - | 4000 | 12000 | 8002

8 | ERA | None | fiboAux | None

9 | GOSUB | None | fiboAux | None

10 | PARAM | 8002 | None | None

11 | = | 8000 | None | 4003

12 | + | 4002 | 4003 | 8003

13 | = | 8003 | None | 4001

14 | GOTO | None | 15 | None

15 | == | 4000 | 12000 | 11001

16 | GOTOF | 11001 | 19 | None

17 | = | 12001 | None | 4001

18 | GOTO | None | 19 | None

19 | == | 4000 | 12001 | 11002

20 | GOTOF | 11002 | 23 | None

21 | = | 12001 | None | 4001

22 | GOTO | None | 23 | None

23 | == | 4000 | 12007 | 11003

24 | GOTOF | 11003 | 27 | None

25 | = | 12007 | None | 4001

26 | GOTO | None | 27 | None

27 | ENDPROC | None | None | None

28 | = | 12007 | None | 4007

29 | < | 4007 | 4005 | 11004

30 | GOTOF | 11004 | 39 | None

31 | ERA | None | fiboAux | None

32 | GOSUB | None | fiboAux | None

33 | PARAM | 4007 | None | None

34 | = | 8000 | None | 4006

35 | PRINT | 4006 | None | None

36 | + | 4007 | 12001 | 8005

37 | = | 8005 | None | 4007

38 | GOTO | None | 29 | None

39 | ENDPROC | None | None | None

40 | ERA | None | fibonacci | None

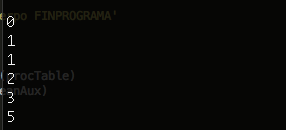
41 | GOSUB | None | fibonacci | None

42 | PARAM | 12011 | None | None

43 | = | 8004 | None | 4008

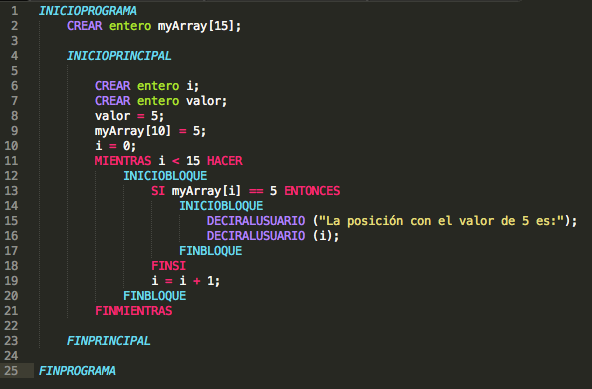
44 | ENDPROG | None | None | None

### Resultado:



## Buscar un elemento en un arreglo

### Código spy



### Cuádruplos

0 | = | 12000 | None | 4001

1 | VER | 12001 | 0 | 14

2 | OFST | 12001 | 0 | 8000

3 | ARYAS | 12000 | 12001 | 8000

4 | = | 12003 | None | 4000

5 | < | 4000 | 12004 | 11000

6 | GOTOF | 11000 | 17 | None

7 | VER | 4000 | 0 | 14

8 | ARYCA | 4000 | 0 | 8001

9 | == | 8001 | 12000 | 11001

10 | GOTOF | 11001 | 14 | None

11 | PRINT | 14000 | None | None

12 | PRINT | 4000 | None | None

13 | GOTO | None | 14 | None

14 | + | 4000 | 12006 | 8002

15 | = | 8002 | None | 4000

16 | GOTO | None | 5 | None

17 | ENDPROG | None | None | None

### Resultado:

Screen Shot 2015-05-04 at 9.50.24 PM.png

## Ordenar un arreglo

### Código spy:



### Cuádruplos:

0 | = | 15000 | None | 7000

1 | VER | 12000 | 0 | 4

2 | OFST | 12000 | 4000 | 8000

3 | ARYAS | 12001 | 12000 | 8000

4 | VER | 12002 | 0 | 4

5 | OFST | 12002 | 4000 | 8001

6 | ARYAS | 12003 | 12002 | 8001

7 | VER | 12004 | 0 | 4

8 | OFST | 12004 | 4000 | 8002

9 | ARYAS | 12005 | 12004 | 8002

10 | VER | 12003 | 0 | 4

11 | OFST | 12003 | 4000 | 8003

12 | ARYAS | 12002 | 12003 | 8003

13 | VER | 12005 | 0 | 4

14 | OFST | 12005 | 4000 | 8004

15 | ARYAS | 12004 | 12005 | 8004

16 | = | 12000 | None | 4006

17 | == | 7000 | 15000 | 11000

18 | GOTOF | 11000 | 49 | None

19 | = | 15002 | None | 7000

20 | = | 12000 | None | 4006

21 | < | 4006 | 12005 | 11001

22 | GOTOF | 11001 | 48 | None

23 | VER | 4006 | 0 | 4

24 | ARYCA | 4006 | 4000 | 8005

25 | + | 4006 | 12002 | 8006

26 | VER | 8006 | 0 | 4

27 | ARYCA | 8006 | 4000 | 8007

28 | > | 8005 | 8007 | 11002

29 | GOTOF | 11002 | 45 | None

30 | VER | 4006 | 0 | 4

31 | ARYCA | 4006 | 4000 | 8008

32 | = | 8008 | None | 4007

33 | + | 4006 | 12002 | 8009

34 | VER | 8009 | 0 | 4

35 | ARYCA | 8009 | 4000 | 8010

36 | VER | 4006 | 0 | 4

37 | OFST | 4006 | 4000 | 8011

38 | ARYAS | 8010 | 4006 | 8011

39 | + | 4006 | 12002 | 8012

40 | VER | 8012 | 0 | 4

41 | OFST | 8012 | 4000 | 8013

42 | ARYAS | 4007 | 8012 | 8013

43 | = | 15000 | None | 7000

44 | GOTO | None | 45 | None

45 | + | 4006 | 12002 | 8014

46 | = | 8014 | None | 4006

47 | GOTO | None | 21 | None

48 | GOTO | None | 17 | None

49 | = | 12000 | None | 4006

50 | < | 4006 | 12001 | 11003

51 | GOTOF | 11003 | 58 | None

52 | VER | 4006 | 0 | 4

53 | ARYCA | 4006 | 4000 | 8015

54 | PRINT | 8015 | None | None

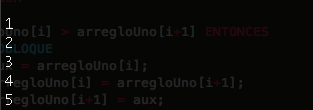
55 | + | 4006 | 12002 | 8016

56 | = | 8016 | None | 4006

57 | GOTO | None | 50 | None

58 | ENDPROG | None | None | None

### Resultado:



## Una multiplicación de arreglos

### Código spy:



### Cuádruplos:

0 | = | 12000 | None | 4013

1 | = | 12000 | None | 4012

2 | < | 4012 | 12002 | 11000

3 | GOTOF | 11000 | 15 | None

4 | \* | 4012 | 12003 | 8000

5 | VER | 4012 | 0 | 4

6 | OFST | 4012 | 4000 | 8001

7 | ARYAS | 8000 | 4012 | 8001

8 | \* | 4012 | 12003 | 8002

9 | VER | 4012 | 0 | 4

10 | OFST | 4012 | 4006 | 8003

11 | ARYAS | 8002 | 4012 | 8003

12 | + | 4012 | 12003 | 8004

13 | = | 8004 | None | 4012

14 | GOTO | None | 2 | None

15 | = | 12000 | None | 4012

16 | < | 4012 | 12002 | 11001

17 | GOTOF | 11001 | 28 | None

18 | VER | 4012 | 0 | 4

19 | ARYCA | 4012 | 4000 | 8005

20 | VER | 4012 | 0 | 4

21 | ARYCA | 4012 | 4006 | 8006

22 | \* | 8005 | 8006 | 8007

23 | + | 8007 | 4013 | 8008

24 | = | 8008 | None | 4013

25 | + | 4012 | 12003 | 8009

26 | = | 8009 | None | 4012

27 | GOTO | None | 16 | None

28 | PRINT | 4013 | None | None

29 | ENDPROG | None | None | None

### Resultado:

Screen Shot 2015-05-04 at 10.04.45 PM.png

## Dibujar figuras geométricas y fractal

### Código spy:





### Cuádruplos:

0 | = | 12000 | None | 4001

1 | > | 4001 | 12001 | 11000

2 | GOTOF | 11000 | 10 | None

3 | MOVER | 12002 | None | None

4 | GIRARDERECHA | 12002 | None | None

5 | MOVER | 12002 | None | None

6 | GIRARDERECHA | 12002 | None | None

7 | - | 4001 | 12002 | 8001

8 | = | 8001 | None | 4001

9 | GOTO | None | 1 | None

10 | ENDPROC | None | None | None

11 | = | 12008 | None | 4003

12 | > | 4003 | 12001 | 11001

13 | GOTOF | 11001 | 19 | None

14 | MOVER | 4002 | None | None

15 | GIRARDERECHA | 12010 | None | None

16 | - | 4003 | 12002 | 8003

17 | = | 8003 | None | 4003

18 | GOTO | None | 12 | None

19 | ENDPROC | None | None | None

20 | == | 4004 | 12001 | 11002

21 | GOTOF | 11002 | 23 | None

22 | GOTO | None | 32 | None

23 | MOVER | 12014 | None | None

24 | GIRARDERECHA | 12010 | None | None

25 | - | 4004 | 12002 | 8005

26 | = | 8005 | None | 4004

27 | PRINT | 4004 | None | None

28 | ERA | None | trianguloRecursivo | None

29 | GOSUB | None | trianguloRecursivo | None

30 | PARAM | 4004 | None | None

31 | PARAM | 4005 | None | None

32 | ENDPROC | None | None | None

33 | = | 12018 | None | 4008

34 | > | 4008 | 12001 | 11003

35 | GOTOF | 11003 | 41 | None

36 | MOVER | 4007 | None | None

37 | GIRARDERECHA | 12020 | None | None

38 | - | 4008 | 12002 | 8007

39 | = | 8007 | None | 4008

40 | GOTO | None | 34 | None

41 | ENDPROC | None | None | None

42 | = | 12023 | None | 4010

43 | DESPINTAR | None | None | None

44 | GIRARIZQUIERDA | 12000 | None | None

45 | MOVER | 12025 | None | None

46 | PINTAR | None | None | None

47 | > | 4010 | 12001 | 11004

48 | GOTOF | 11004 | 54 | None

49 | MOVER | 12014 | None | None

50 | GIRARDERECHA | 12028 | None | None

51 | - | 4010 | 12002 | 8009

52 | = | 8009 | None | 4010

53 | GOTO | None | 47 | None

54 | ENDPROC | None | None | None

55 | < | 4011 | 12002 | 11005

56 | GOTOF | 11005 | 58 | None

57 | GOTO | None | 88 | None

58 | GIRARDERECHA | 13000 | None | None

59 | - | 4011 | 12002 | 8011

60 | = | 8011 | None | 4011

61 | ERA | None | piramide | None

62 | GOSUB | None | piramide | None

63 | PARAM | 4011 | None | None

64 | PARAM | 4012 | None | None

65 | PINTAR | None | None | None

66 | / | 4012 | 12033 | 8012

67 | MOVER | 8012 | None | None

68 | / | 4012 | 12008 | 8013

69 | MOVER | 8013 | None | None

70 | GIRARDERECHA | 12010 | None | None

71 | - | 4011 | 12002 | 8014

72 | = | 8014 | None | 4011

73 | ERA | None | piramide | None

74 | GOSUB | None | piramide | None

75 | PARAM | 4011 | None | None

76 | PARAM | 4012 | None | None

77 | / | 4012 | 12037 | 8015

78 | MOVER | 8015 | None | None

79 | / | 4012 | 12008 | 8016

80 | MOVER | 8016 | None | None

81 | GIRARDERECHA | 12010 | None | None

82 | - | 4011 | 12002 | 8017

83 | = | 8017 | None | 4011

84 | ERA | None | piramide | None

85 | GOSUB | None | piramide | None

86 | PARAM | 4011 | None | None

87 | PARAM | 4012 | None | None

88 | ENDPROC | None | None | None

89 | < | 4014 | 12001 | 11006

90 | GOTOF | 11006 | 93 | None

91 | MOVER | 5000 | None | None

92 | GOTO | None | 120 | None

93 | / | 5000 | 12008 | 9001

94 | - | 4014 | 12002 | 8018

95 | ERA | None | fractalSnowFlake | None

96 | GOSUB | None | fractalSnowFlake | None

97 | PARAM | 8018 | None | None

98 | PARAM | 9001 | None | None

99 | GIRARDERECHA | 12044 | None | None

100 | / | 5000 | 12008 | 9002

101 | - | 4014 | 12002 | 8019

102 | ERA | None | fractalSnowFlake | None

103 | GOSUB | None | fractalSnowFlake | None

104 | PARAM | 8019 | None | None

105 | PARAM | 9002 | None | None

106 | GIRARIZQUIERDA | 12010 | None | None

107 | / | 5000 | 12008 | 9003

108 | - | 4014 | 12002 | 8020

109 | ERA | None | fractalSnowFlake | None

110 | GOSUB | None | fractalSnowFlake | None

111 | PARAM | 8020 | None | None

112 | PARAM | 9003 | None | None

113 | GIRARDERECHA | 12044 | None | None

114 | / | 5000 | 12008 | 9004

115 | - | 4014 | 12002 | 8021

116 | ERA | None | fractalSnowFlake | None

117 | GOSUB | None | fractalSnowFlake | None

118 | PARAM | 8021 | None | None

119 | PARAM | 9004 | None | None

120 | ENDPROC | None | None | None

121 | INPUT | entero | None | 8022

122 | = | 8022 | None | 4015

123 | == | 4015 | 12002 | 11007

124 | GOTOF | 11007 | 129 | None

125 | ERA | None | circulo | None

126 | GOSUB | None | circulo | None

127 | PARAM | 12002 | None | None

128 | GOTO | None | 129 | None

129 | == | 4015 | 12033 | 11008

130 | GOTOF | 11008 | 135 | None

131 | ERA | None | triangulo | None

132 | GOSUB | None | triangulo | None

133 | PARAM | 12028 | None | None

134 | GOTO | None | 135 | None

135 | == | 4015 | 12054 | 11009

136 | GOTOF | 11009 | 142 | None

137 | ERA | None | trianguloRecursivo | None

138 | GOSUB | None | trianguloRecursivo | None

139 | PARAM | 12008 | None | None

140 | PARAM | 12028 | None | None

141 | GOTO | None | 142 | None

142 | == | 4015 | 12008 | 11010

143 | GOTOF | 11010 | 148 | None

144 | ERA | None | pentagono | None

145 | GOSUB | None | pentagono | None

146 | PARAM | 12028 | None | None

147 | GOTO | None | 148 | None

148 | == | 4015 | 12037 | 11011

149 | GOTOF | 11011 | 154 | None

150 | ERA | None | fractalEstrella | None

151 | GOSUB | None | fractalEstrella | None

152 | PARAM | 12028 | None | None

153 | GOTO | None | 154 | None

154 | == | 4015 | 12018 | 11012

155 | GOTOF | 11012 | 161 | None

156 | ERA | None | piramide | None

157 | GOSUB | None | piramide | None

158 | PARAM | 12008 | None | None

159 | PARAM | 12062 | None | None

160 | GOTO | None | 161 | None

161 | == | 4015 | 12064 | 11013

162 | GOTOF | 11013 | 168 | None

163 | ERA | None | fractalSnowFlake | None

164 | GOSUB | None | fractalSnowFlake | None

165 | PARAM | 12037 | None | None

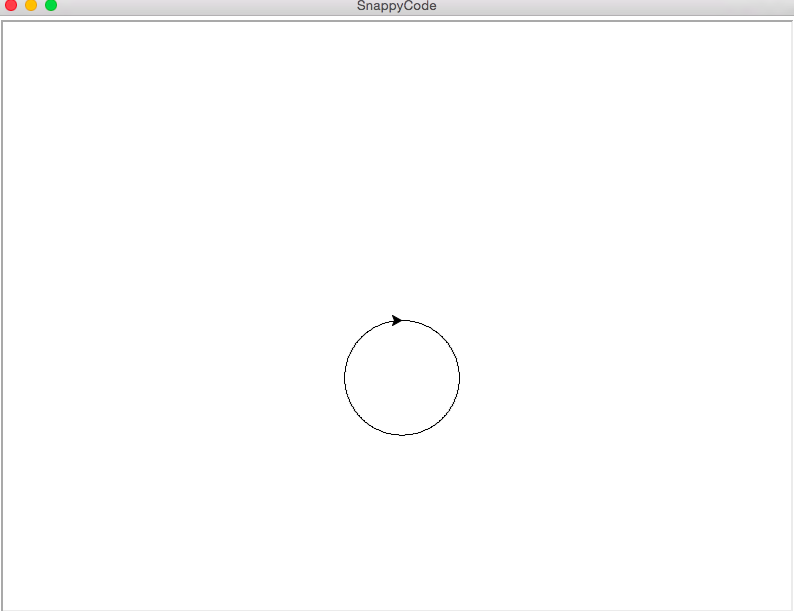
166 | PARAM | 13006 | None | None

167 | GOTO | None | 168 | None

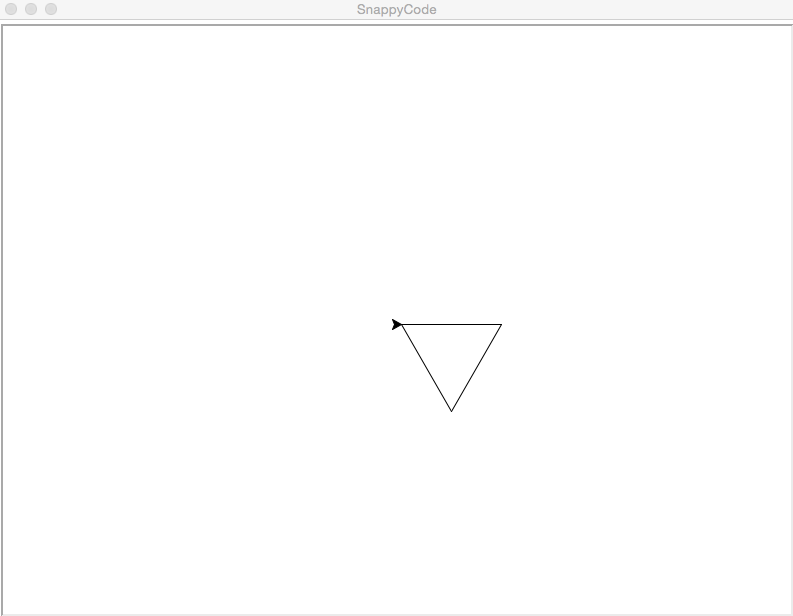
168 | ENDPROG | None | None | None

### Resultado:

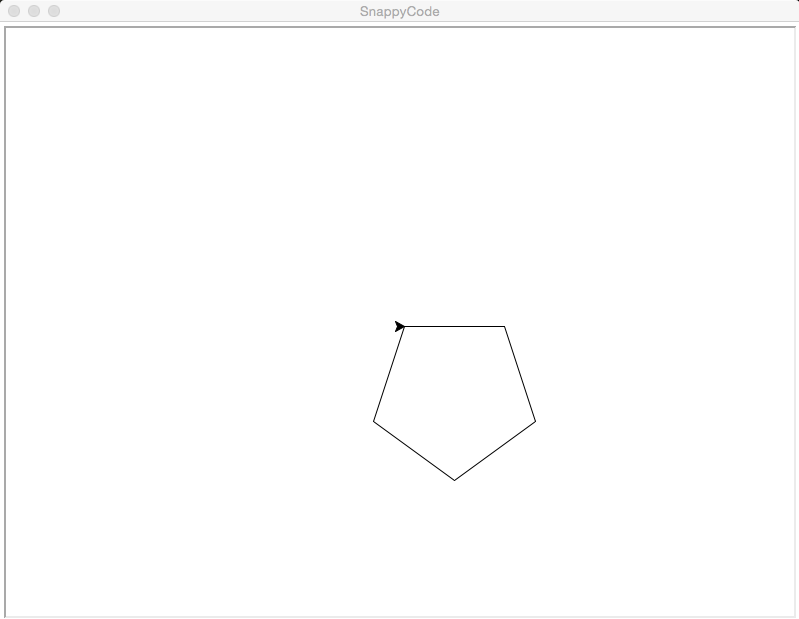
Círculo



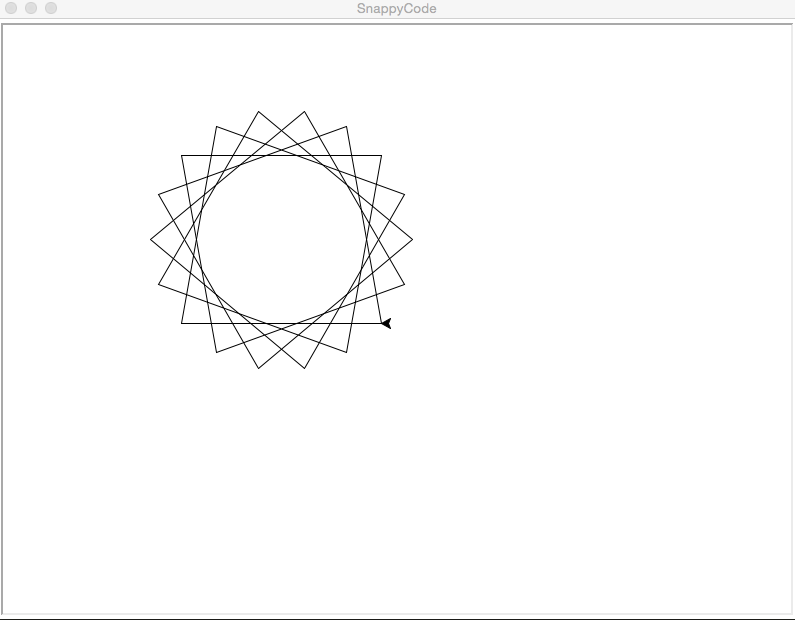
Triángulo



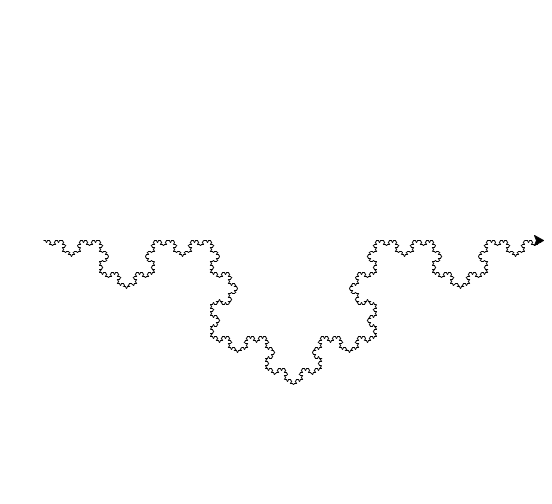
Pentágono



Estrella



Fractal



# ANEXO 1: Manual de Usuario

Ejecución del lenguaje

Para poder ejecutar el lenguaje es necesario seguir los siguientes pasos:

* Contar con una versión de Python superior a la 3.0+
* Tener las librerías del compilador que se pueden descargar de la siguiente liga <https://github.com/jplarar/SnappyCode>
* Tener nuestro archivo .spy con el código escrito en el lenguaje SnappyCode

Compelido con todos los requerimientos anteriores se ejecuta de la siguiente.

Windows

Abrir CMD y navegar hasta la carpeta en donde se encuentran las librerías SnappyCode descargadas.

Ejecutar el siguiente comando, cambiando “nombreDelArchivo.spy” por el nombre de nuestro archivo a compilar.

|  |
| --- |
| $python snappycodeParse.py nombreDelArchivo.spy |

Mac OS X

Abrir la terminal y navegar hasta la carpeta en donde se encuentran las librerías de SnappyCode descargadas.

Ejecutar el siguiente comando, cambiando “nombreDelArchivo.spy” por el nombre de nuestro archivo a compilar.

|  |
| --- |
| $python3 snappycodeParse.py nombreDelArchivo.spy |

Estructura básica del lenguaje

Crear un programa

Todo programa escrito en el lenguaje SnappyCode debe seguir una estructura establecida, empezando por el inicio y fin del programa

|  |
| --- |
| INICIOPROGRAMA  FINPROGRAMA |

Dentro de estos debe incluir también la función principal que es la que ejecuta el programa en automático.

|  |
| --- |
| INICIOPROGRAMA  INICIOPRINCIPAL  FINPRINCIPAL  FINPROGRAMA |

Funciones

Las funciones en este lenguaje tienen que ser declaradas antes de que sean llamadas.

Estas siguen el siguen un formato de creación parecido a la función inicio programa, soportan el paso de parámetros y obligatoriamente tienen que regresar un valor el cual debe ser igual al tipo de con el cual se declaró la función.

|  |
| --- |
| INICIOPROGRAMA  INICIOFUNCION entero nombreDeFuncion PARAMETROS entero param1  REGRESA 1;  FINFUNCION  INICIOPRINCIPAL  FINPRINCIPAL  FINPROGRAMA |

Las funciones pueden ser llamadas dentro de otras funciones o dentro de la función principal de la siguiente manera:

|  |
| --- |
| nombreDeFuncion(100); |

Donde se especifica el nombre de la función a la cual se hace referencia y después se escribe los parámetros que se mandan para la ejecución de la función

Variables

SnappyCode soporta la creación y uso de variables del tipo booleano, entero, flotante y texto.

Creación

Para crear una variable de cualquiera de los tipos antes mencionados se hace en el siguiente formato se pone la palabra crear, seguida del tipo y por último el nombre de la variable seguida de un punto y coma.

|  |
| --- |
| crear entero variableEntera;  crear flotante variableFlotante;  crear booleano variableBool;  crear texto variableTexto; |

Las variables que sean creadas afuera de una función e inmediatamente después del estatuto INICIOPROGRAMA se almacenan como variables globales.

Asignación

Para asignar o guardar algún valor en una variable está tiene que haber sido creada anteriormente.

El formato que debe seguir es el siguiente: nombre de la variable seguida de un igual y después una expresión que puede ser otra variable o algún valor u operación.

|  |
| --- |
| crear entero variableEntera;  variableEntera = 10;  variableEntera = 5 \* 5; |

## Operadores

### Matemático

SnappyCode te permite hacer cálculos con valores, permitiendo así hacer las principales operaciones aritméticas como:

* +: Sumar dos objetos juntos. 1 + 1 es 2.
* -: Restar el segundo número al primero 3 - 2 es 1.
* /: Dividir dos números. El resultado será un número flotante. 1 / 2 es 0.5.
* \*: Multiplicar el operando izquierdo con el de la derecha. 2 \* 2 es 4.
* (): Estos son utilizados para dar prioridad a las operaciones.

### Lógicos

Se pueden combinar múltiples expresiones con los siguientes operadores:

* &&: Regresa verdadero si ambos tanto izquierdos como derechos son verdaderos.
* ||: Regresa verdadero si el lado izquierdo o derecho es verdadero.

### Comparación

Los siguientes operadores de comparación son soportados en cualquier expresión: ==, !=, <, >, >=, and <=.

Control de flujo

Condiciones

El lenguaje SnappyCode maneja un estatuto condicional mejor conocido como if, el cual lleva el siguiente formato:

|  |
| --- |
| SI 5 < 10 ENTONCES  INICIOBLOQUE    FINBLOQUE  FINSI |

Comenzando por la palabra SI seguida de una expresión la cual al ser verdadera ejecuta el código de la parte que se encuentra dentro de INICIOBLOQUE y FINBLOQUE de lo contrario no se ejecuta.

También existe lo que se conoce comúnmente como else, que en este caso es SINO siendo utilizado de la siguiente manera:

|  |
| --- |
| SI 5 < 10 ENTONCES  INICIOBLOQUE    FINBLOQUE  SINO  INICIOBLOQUE  FINBLOQUE  FINSI |

Donde se evalúa la expresión seguida del SI, si el resultado de esta expresión es verdadero ejecuta el código dentro de la primer parte, de lo contrario “si no”, ejecuta la segunda parte de código.

Ciclos

SnappyCode cuenta con un estatuto de repetición de código el cual se ejecuta n veces dependiendo del resultado de una expresión.

Este se asemeja a lo que en otros lenguajes es conocido como while.

Sigue una estructura parecida a la de condición:

|  |
| --- |
| MIENTRAS 1 < 2 HACER  INICIOBLOQUE  FINBLOQUE  FINMIENTRAS |

En este caso se ejecuta el código que se encuentra en la parte entre el INICIOBLOQUE y FINBLOQUE en caso de que la expresión sea verdadero y se sigue ejecutando hasta que la expresión sea falsa.

Hay que tener cuidado con esta expresión ya que puede llegar a ciclar la ejecución del programa.

Estructura de datos

Arreglos

Este lenguaje maneja una forma sencilla de agrupar datos, la cual se hace mediante la estructura de arreglos.

Los arreglos son variables dentro de las cuales se pueden almacenar n cantidad de valores.

Los arreglos se crean de manera similar a las variables, con la variación de que se les asigna un tamaño o cantidad máxima de datos que puede almacenar.

|  |
| --- |
| crear entero miArreglo[150]; |

Se utiliza la palabra crea seguida del tipo de datos que almacena el arreglo, después se le asigna un nombre y por último se le da un tamaño el cual va entre corchete seguido de un punto y coma.

Estatutos especiales

Entrada y salida

Para poder interactuar con el usuario SnappyCode maneja 2 estatutos, uno para la entrada que es PEDIRALUSUARIO y otro para la salida el cual es DECIRALUSUARIO y se utilizan de la siguiente manear:

|  |
| --- |
| DECIRALUSUARIO(“introduce un valor”);  PEDIRALUSUARIO(varValor); |

En el estatuto DECIRALUSUARIO se manda una expresión la cual se desplegará en la consola para que pueda ser vista por el usuario.

El estatuto PEDIRALUSUARIO se queda esperando en la consola hasta que el usuario introduce una dato el cual será almacenado en la variable que se pone entre los paréntesis.

Gráficos

Estos son los estatutos principales del lenguaje los cuales son utilizados para reflejar toda la lógica de programación mediante una salida de gráficos.

Al crear un programa en SnappyCode siempre se crea una ventana en blanco en la cual aparece una flecha que es en donde se reflejan las acciones de los siguientes estatutos.

|  |
| --- |
| GIRARDERECHA 90;  GIRARIZQUIERDA 90;  MOVER 100;  RETROCEDE 100; |

Tanto el estatuto GIRARDERECHA como GIRARIZQUIERDA son utilizados para para girar la flecha como sus nombres lo dicen estas están seguidas de una expresión la cual indica la cantidad en grados que giran.

MOVER y RETROCEDER son utilizadas para trazar líneas rectas ya sea hacia adelante o hacia atrás, éstas van seguidas por una expresión la cual determina la cantidad de pixeles que se mueve el trazo.

También hay estatutos de gráficos que no dependen de la flecha dibujada en el gráfico que es:

|  |
| --- |
| BORRAR |

BORRAR es utilizado para limpiar completamente la pantalla de salida gráfica.

|  |
| --- |
| DESPINTAR |

Es utilizado para mover la flecha con los estatutos de MOVER o RETROCEDER sin que marque el trazo

|  |
| --- |
| PINTAR |

Pintar revierte la acción de DESPINTAR hace que al mover la flecha se vuelva a pintar