Diseño de compiladores

Proyecto: lenguaje Statlang

Eduardo González Torres

2 de octubre de 2022

Objetivo general

Desarrollar un lenguaje que permita realizar un análisis estadístico básico de un conjunto de datos numéricos.

Características del lenguaje

Lista de tokens

Keywords: program, var, int, float, file, if, else, from, to, do, main, func, void, read, write, ;, ,, {, }, (,), [,]

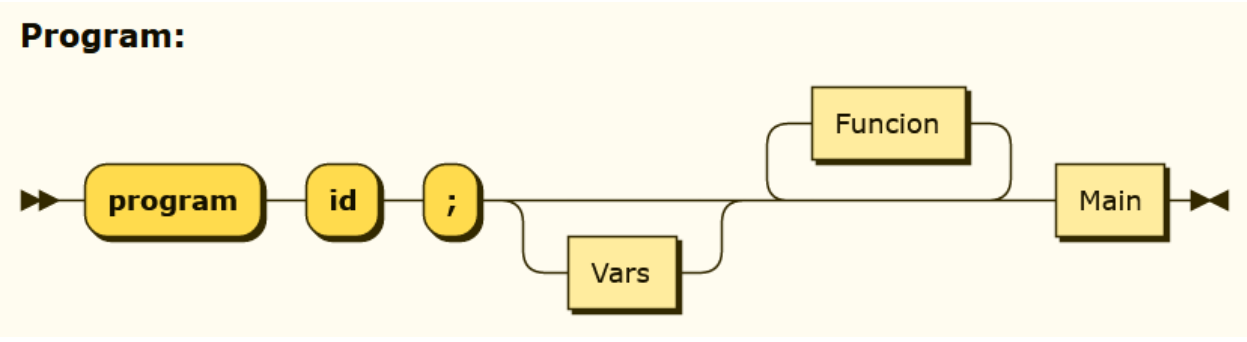
Operadores: =, <, >, <>, ==, +, -, *, /, & (AND), | (OR)

Valores: id, cteI, cteF, cteString

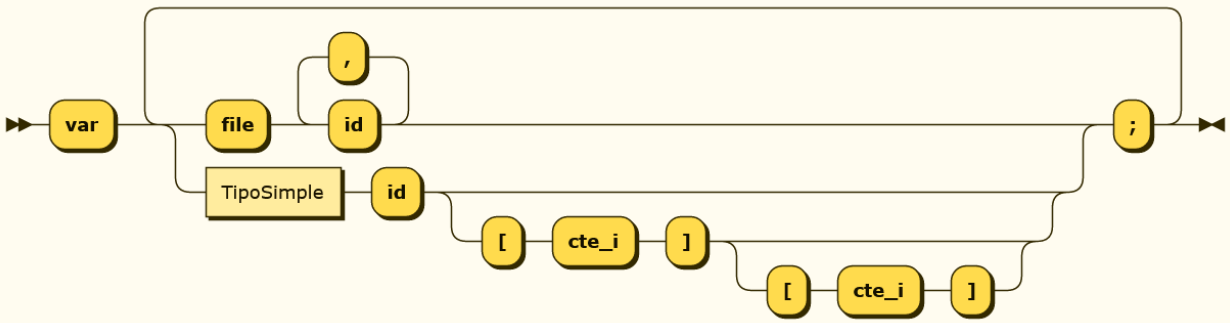
program	->	program
var	->	var
int	->	int
float	->	float
file	->	file
if	->	if
else	->	else
from	->	from
to	->	to
do	->	do
mainStart	->	main
func	->	func
void	->	void
;	->	**;**
,	->	**,**
{	->	\{
}	->	\}

(	->	\(
)	->	\)
[	->	\[
]	->	\]
assignOp	->	=
lessThanOp	->	<
greaterThanOp	->	>
differentOp	->	<>
equalOp	->	==
sumOp	->	\+
subOp	->	-
mulOp	->	*
divOp	->	/
andOp	->	&
orOp	->	|
id	->	[A-Za-z] ([A-Za-z] | [0-9])*
cteI	->	[0-9]+
cteF	->	[0-9]+ \. [0-9]+
cteString	->	“.+”

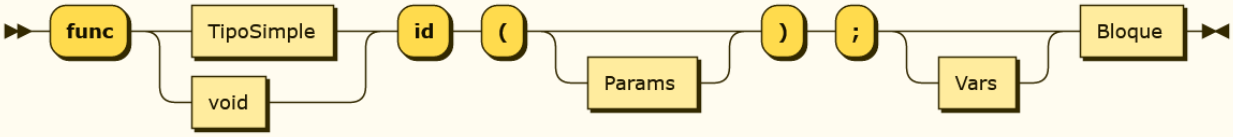
Diagramas de sintaxis



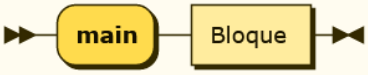
Vars:



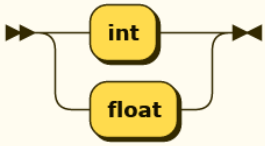
Funcion:



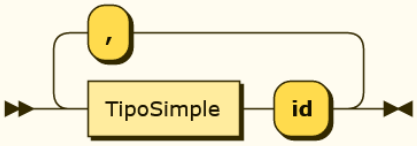
Main:



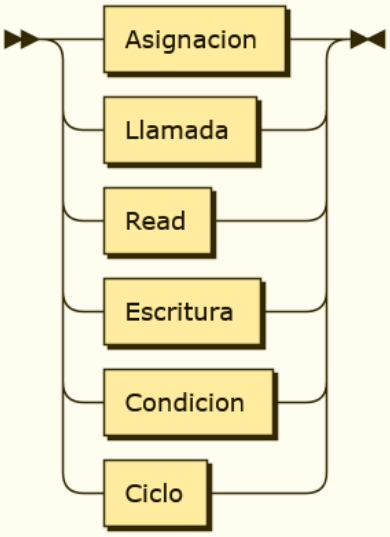
TipoSimple:



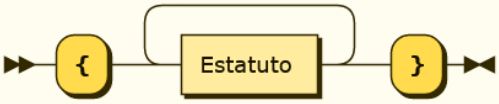
Params:



Estatuto:

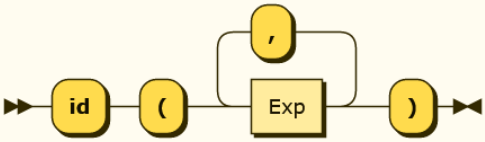


Bloque:

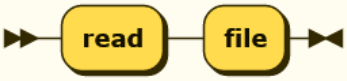


Asignacion:

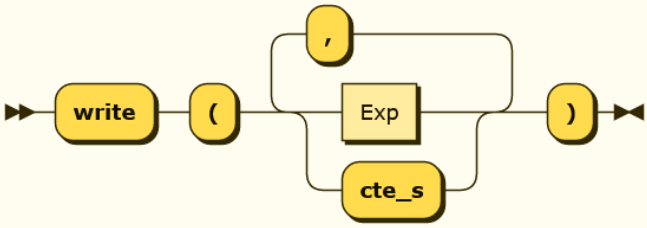
Llamada:



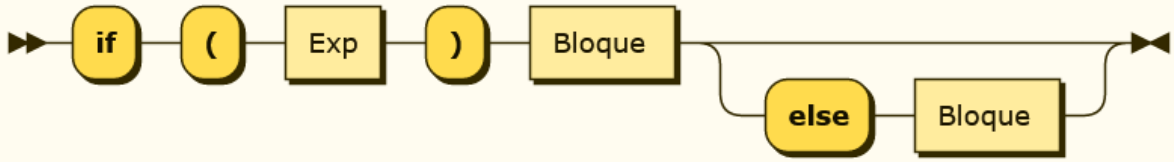
Read:



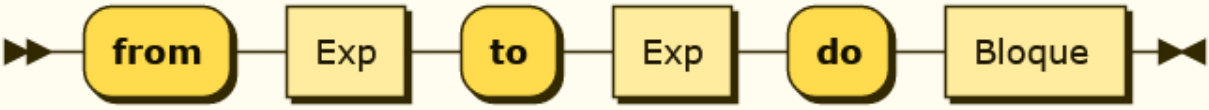
Escritura:



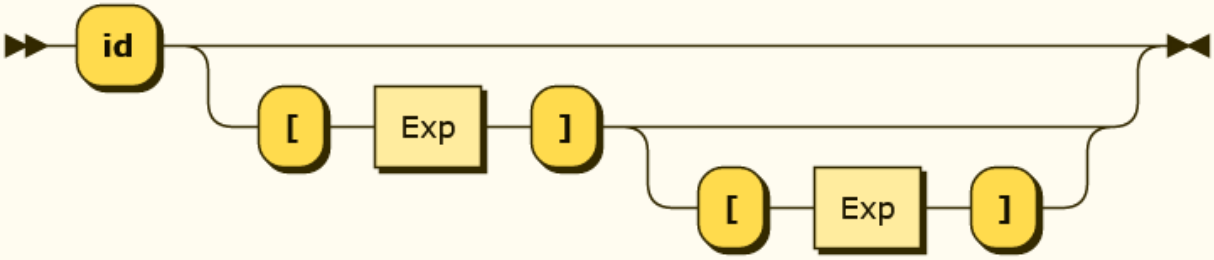
Condicion:



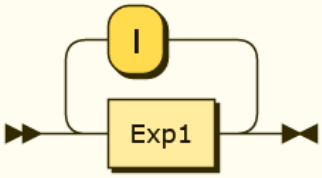
Ciclo:



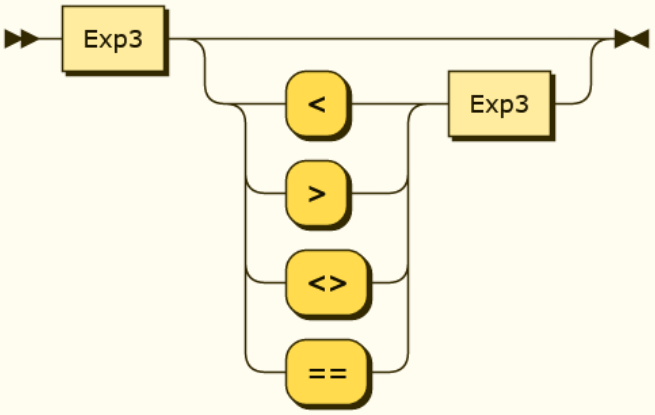
Variable:



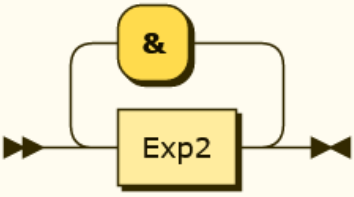
Exp:

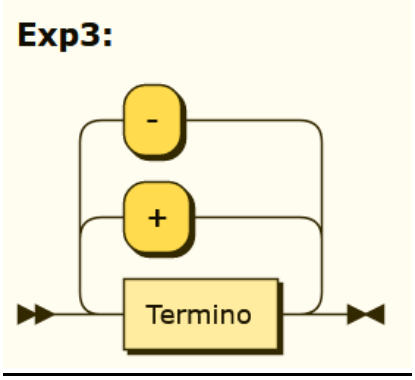


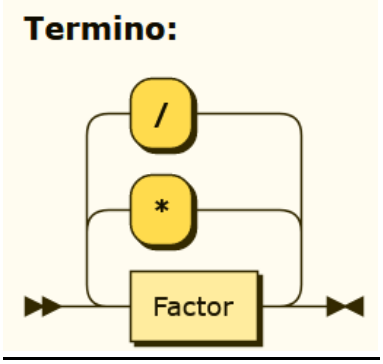
Exp2:

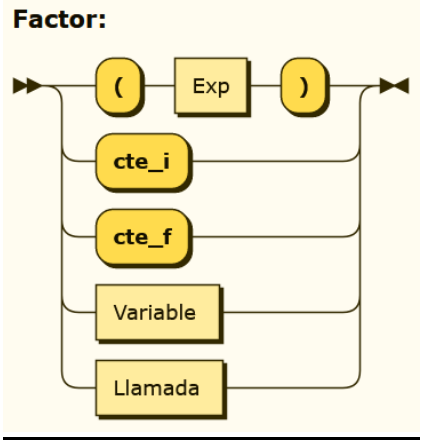


Exp1:









Gramática formal

```
Program     ->    program id ; ProgramA ProgramB Main
ProgramA    ->    Vars | ϵ
ProgramB    ->    Funcion ProgramB | ϵ
Vars        ->    var VarsA
VarsA       ->    VarsB ; VarsF
```

```
VarsB      ->   file id VarsE | TipoSimple id VarsC
VarsC      ->   [ cte_i ] VarsD | ϵ
VarsD      ->   [ cte_i ] | ϵ
VarsE      ->   , id VarsE | ϵ
VarsF      ->   VarsA | ϵ
Funcion    ->   func FuncionA id ( FuncionB ) ; FuncionC Bloque
FuncionA   ->   TipoSimple | void
FuncionB   ->   Params | ϵ
FuncionC   ->   Vars | ϵ
Main       ->   main Bloque
TipoSimple ->   int | float
Params     ->   TipoSimple id ParamsA
ParamsA    ->   , TipoSimple id ParamsA | ϵ
Bloque     ->   { BloqueA }
BloqueA    ->   Estatuto BloqueA | ϵ
Estatuto   ->   Asignacion | Llamada | Read | Escritura | Condicion | Ciclo
Asignacion ->   Variable = Exp
Llamada    ->   id ( Exp LlamadaA )
LlamadaA   ->   , Exp LlamadaA | ϵ
Read       ->   read id
Escritura  ->   write ( EscrituraA EscrituraB )
EscrituraA ->   Exp | cte_s
EscrituraB ->   , EscrituraA EscrituraB | ϵ
Condicion  ->   if ( Exp ) Bloque CondicionA
CondicionA ->   else Bloque | ϵ
Ciclo      ->   from Exp to Exp do Bloque
Variable   ->   id VariableA
VariableA  ->   [ Exp ] VariableB | ϵ
VariableB  ->   [ Exp ] | ϵ
Exp        ->   Exp1 ExpA
```

```
ExpA        ->    | Exp1 ExpA | ϵ
Exp1        ->    Exp2 Exp1A
Exp1A       ->    & Exp2 Exp1A | ϵ
Exp2        ->    Exp3 Exp2A
Exp2A       ->    Exp2B Exp3 | ϵ
Exp2B       ->    < | > | <> | ==
Exp3        ->    Termino Exp3A
Exp3A       ->    Exp3B Termino Exp3A | ϵ
Exp3B       ->    + | -
Termino     ->    Factor TerminoA
TerminoA    ->    TerminoB Factor TerminoA | ϵ
TerminoB    ->    * | /
Factor      ->    ( Exp ) | cte_i | cte_f | Variable | Llamada
```

Características semánticas principales

Se pueden declarar variables globales, así como variables locales a una función o al main.

Pueden declararse arreglos de 1 o 2 dimensiones, de un tamaño máximo de 100 elementos por dimensión.

Los parámetros de una función únicamente son de entrada, y el tipo de retorno de una función puede ser cualquier tipo simple soportado por el lenguaje, o void.

En cuanto a los operadores, se usan las prioridades tradicionales, y pueden usarse paréntesis para alterar las prioridades.

La lógica usada será numérica; cualquier número que no sea 0 será considerado como *true*, y 0 será *false*.

Funciones especiales

Este lenguaje permitirá realizar un análisis estadístico simple de datos numéricos contenidos en un archivo de texto. El archivo de texto deberá tener el siguiente formato:

```
%% Los datos son ints o floats
```

```
%% Cada dato separado por un espacio, cada renglón por un newline
dato11      dato21      dato31      dato41
dato12      dato22      dato32      dato42
%% Continúa…
```

El archivo podrá contener un máximo de 100 columnas, y 100 datos por columna. Se incluirá una función que genera una matriz con los datos del archivo. Además, el lenguaje proveerá un conjunto de funciones estadísticas básicas que se podrán aplicar sobre esa matriz, en particular:

- Ordenar los datos de mayor a menor y viceversa
- Calcular media, mediana, moda, rango, varianza, sesgo, curtosis y percentiles
- Visualizar los datos en un histograma y un diagrama de caja

Tipos de datos

Los tipos de datos soportados por este lenguaje son identificadores, *keywords*, arreglos de 1 o 2 dimensiones, ints, floats, *file*s (un tipo no atómico que sirve para cargar los datos del archivo a una representación matricial) constantes enteras, constantes flotantes y constantes string (letreros).

Lenguaje y OS usados para el desarrollo

Se usará Python en Windows 10.

Bibliografía

Documento de descripción de lenguaje DS--

Archivos en la carpeta de apoyos de clase en Google Drive