UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

ESCUELA DE SISTEMAS

ORGANIZACIÓN DE LENGUAJES COMPILADORES

ING. MARIO BAUTISTA

USUARIO

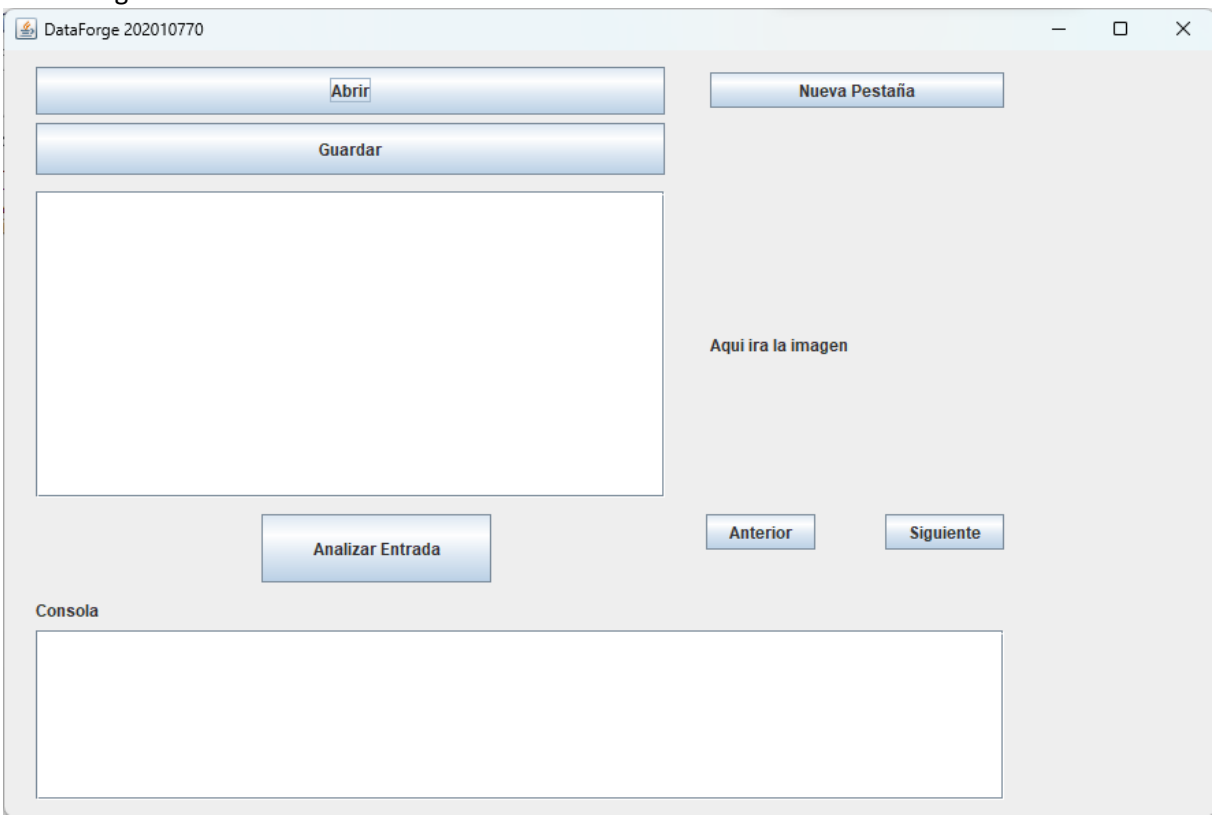
LUIS MARIANO MOREIRA GARCÍA

202010770

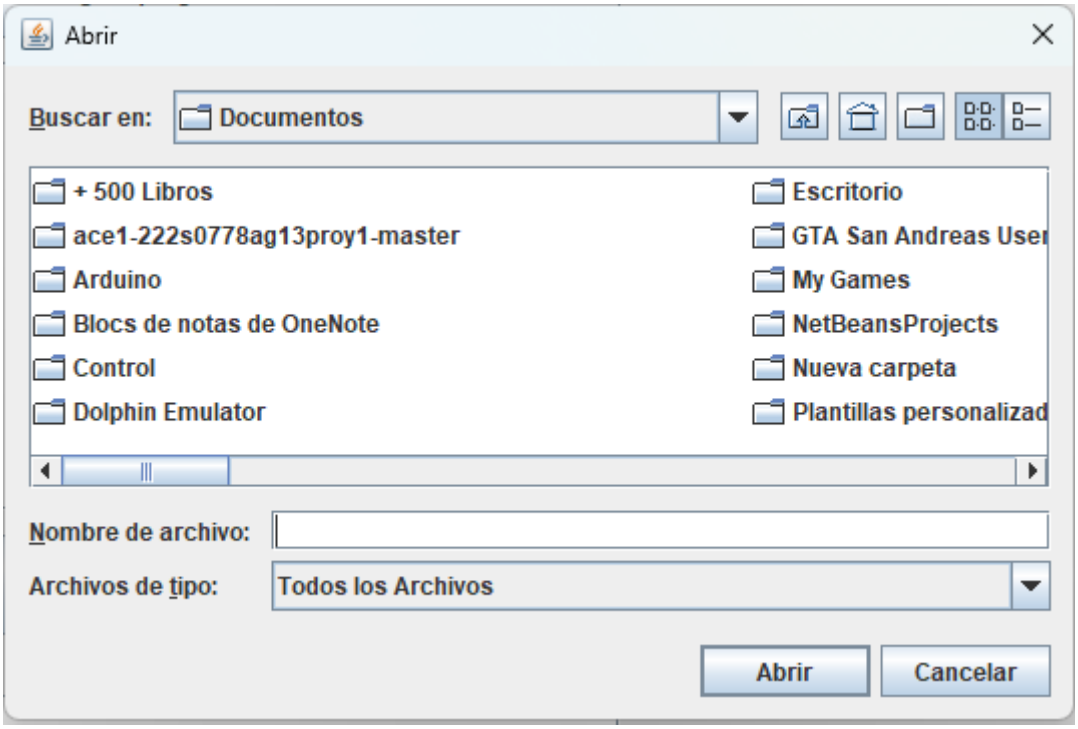
REQUISITOS MINIMOS:

- 100mb de ram
- Intel Pentium

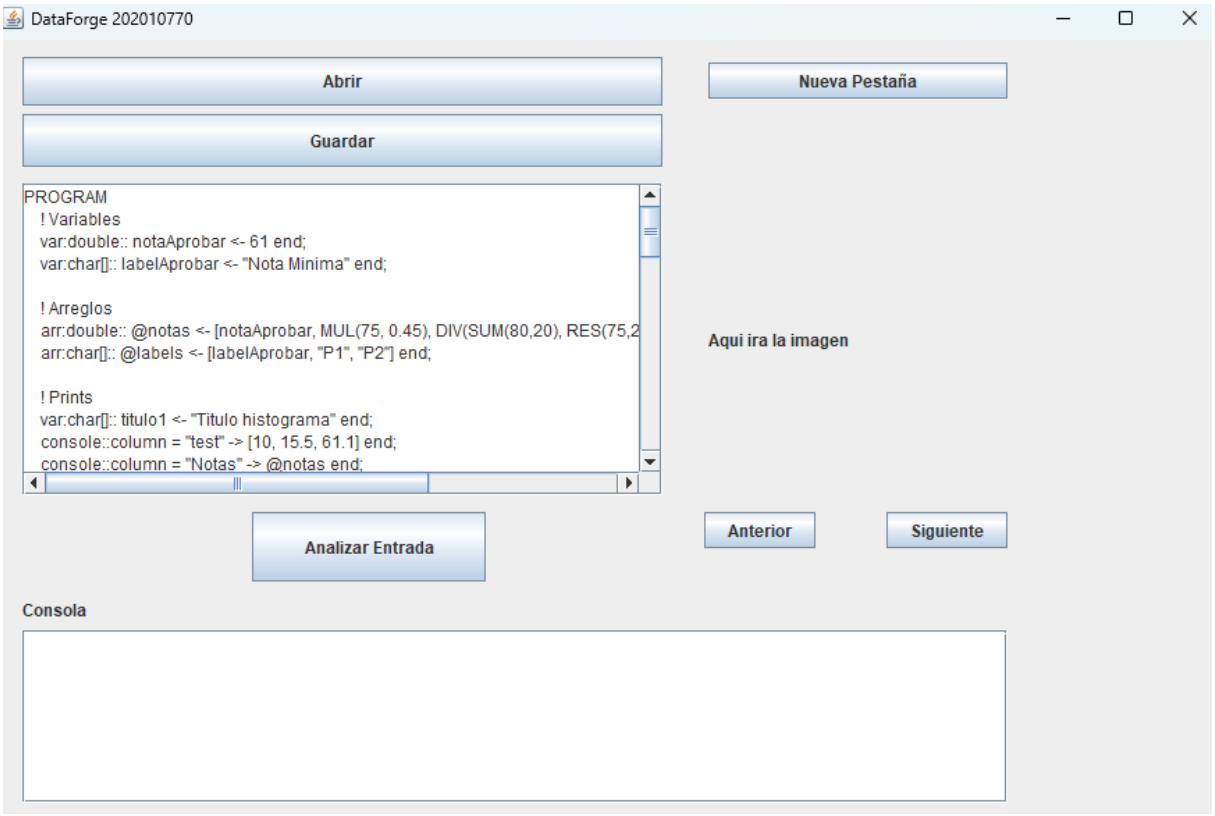
Interfaz grafica:



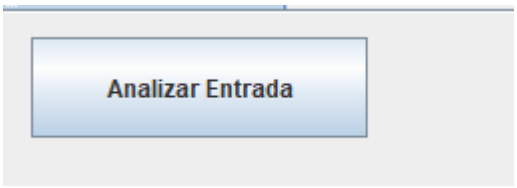
Si se selecciona abrir se desplegara un file chooser y se podrá escoger cualquier archivo con extensión “.df”, igual no es ningún empedimiento abrir otra clase de archivos.



Al seleccionar un archivo:



Para generar un archivo:



Al analizar se generan los archivos:

Reporte Lista de Elementos

Variable: char_temp

Tipo	Variable	Dato

Variable: double_temp

Tipo	Variable	Dato
double	char_temp	2
double	char_temp	2
double	char_temp	2
double	char_temp	5
double	char_temp	5
double	char_temp	7
double	char_temp	8

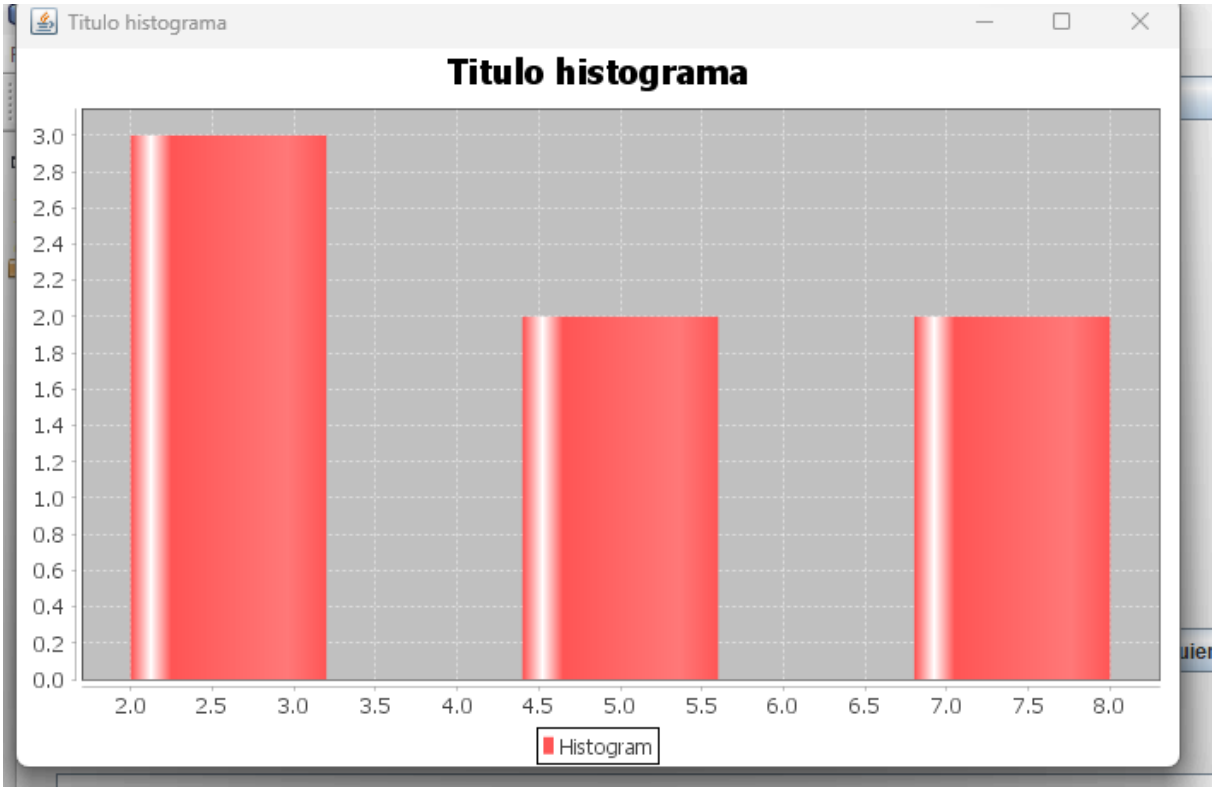
Variable: notas

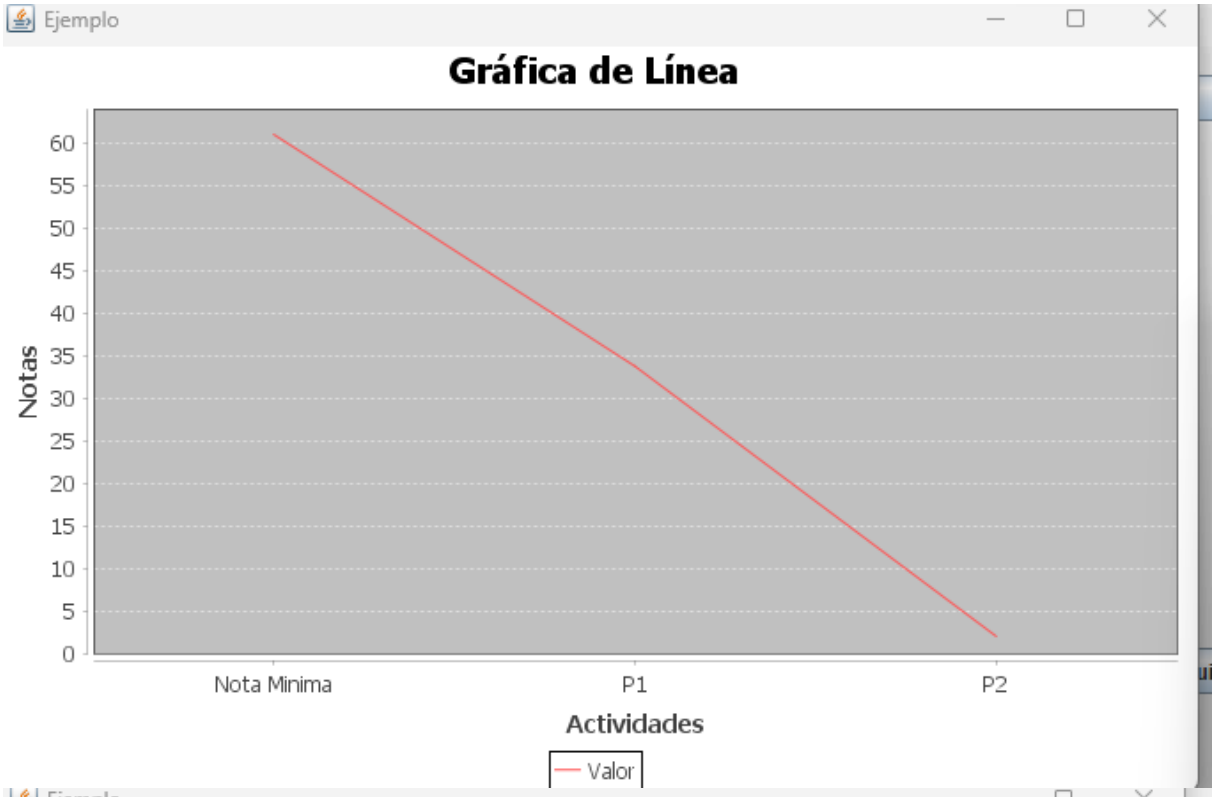
Tipo	Variable	Dato
double	notas	61.0
double	notas	33.75
double	notas	2.0

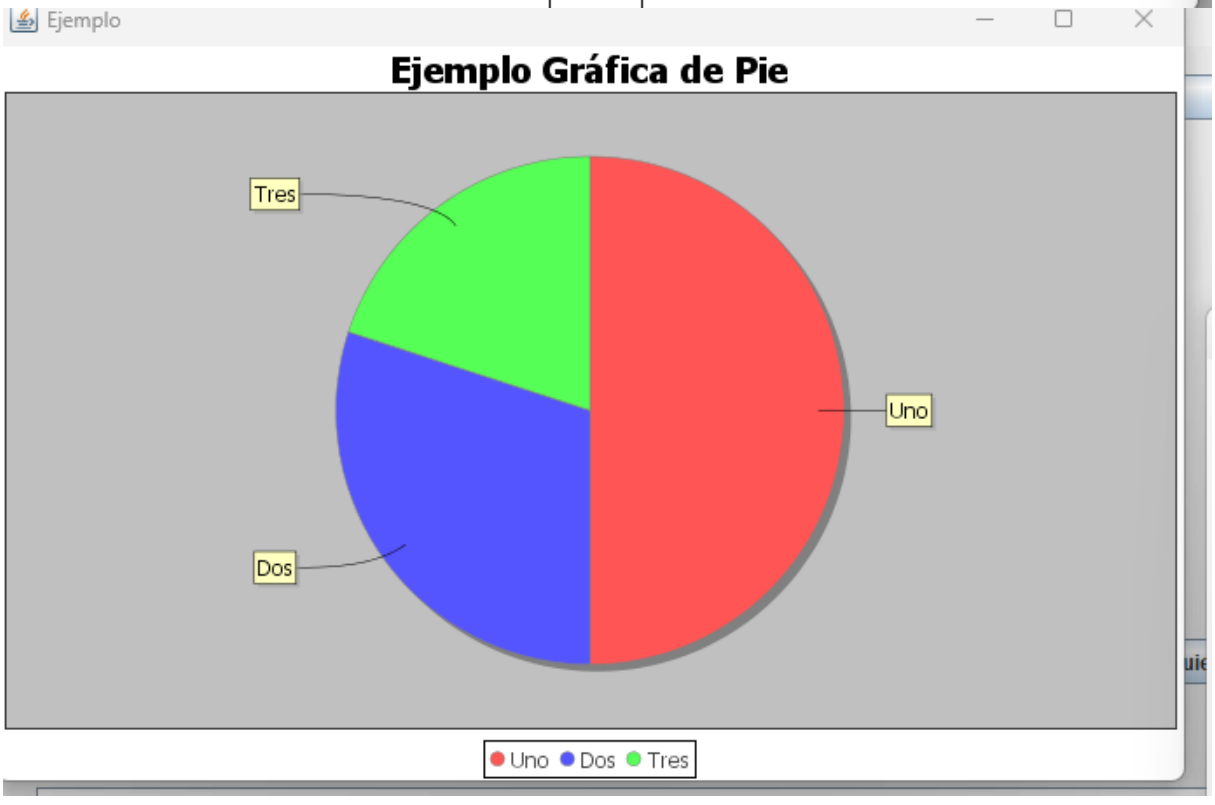
Variable: labels

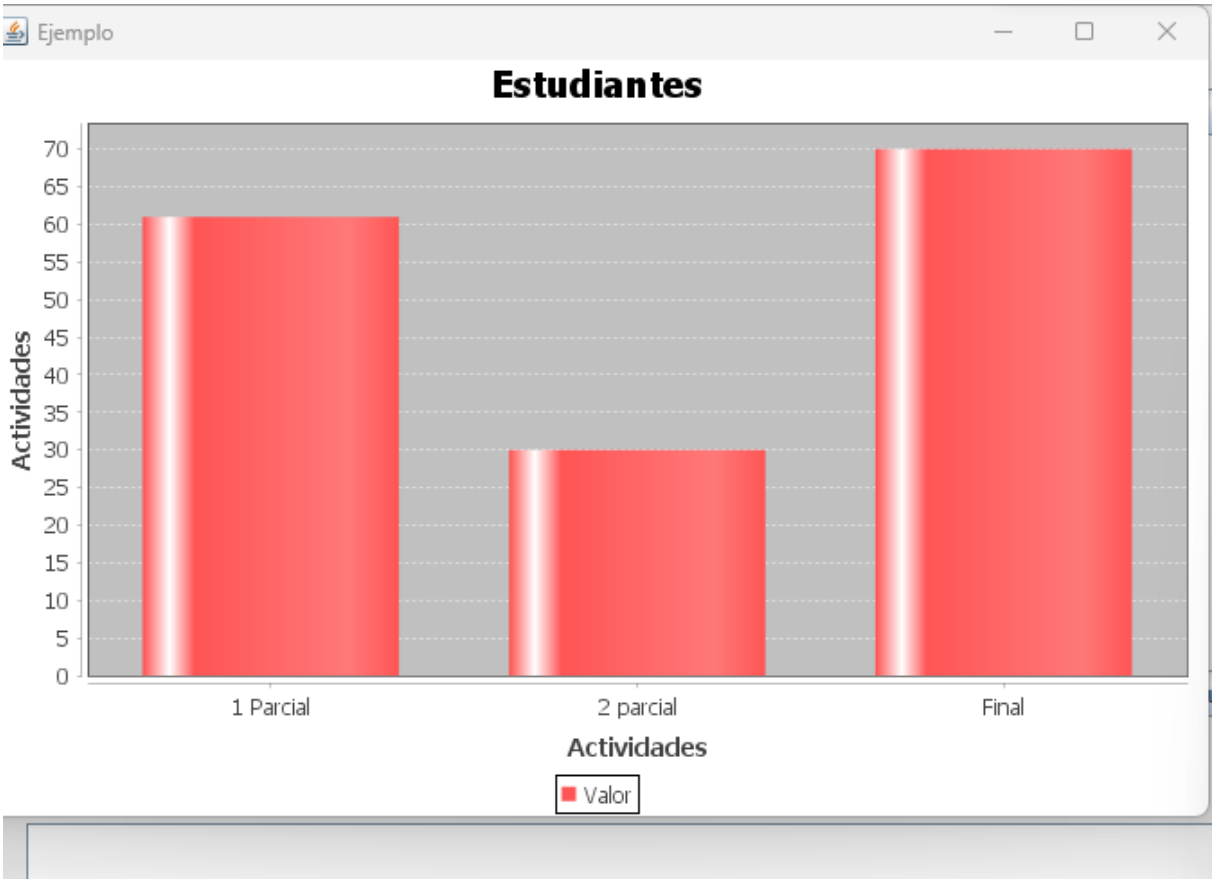
Reporte Tabla de Simbolos

Tipo	Variable	Dato
double	notaAprobar	61
cadena	labelAprobar	Nota Minima
cadena	titulo1	Titulo histograma
double	gb1	61
cadena	gbt	Datos


Titulo histograma
Titulo histograma
3.0
2.8
2.6
2.4
2.2
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
Histogram


Ejemplo
Gráfica de Línea
Notas
60
55
50
45
40
35
30
25
20
15
10
5
0
Nota Minima
P1
P2
Actividades
Valor


Ejemplo
Ejemplo Gráfica de Pie
Tres
Uno
Dos
Uno
Dos
Tres



Salida de consola: (Se necesita tener la verdadera consola para ver)

```
----------
test
----------
----------
Notas
----------
61.0
33.75
2.0
----------
Titulo histograma
----------
Nota Minima
P1
P2
Media, Mediana, Moda, Varianza, Max, Min
32.25, 33.75, 33.75, 581.2916666666666, 61.0, 2.0
Hola Mundo, 1.0, 61, Nota Minima
```

A su vez que el registro de errores y de tokens se pueden ver arriba:

```
Número de línea: 49, Número de columna: 25, Lexema: ;
Número de línea: 50, Número de columna: 4, Lexema: )
Número de línea: 50, Número de columna: 6, Lexema: end
Número de línea: 50, Número de columna: 9, Lexema: ;
Número de línea: 52, Número de columna: 4, Lexema: Histogram
Número de línea: 52, Número de columna: 13, Lexema: (
Número de línea: 53, Número de columna: 8, Lexema: titulo
Número de línea: 53, Número de columna: 14, Lexema: :
Número de línea: 53, Número de columna: 15, Lexema: :
Número de línea: 53, Número de columna: 16, Lexema: char[]
Número de línea: 53, Número de columna: 23, Lexema: =
Número de línea: 53, Número de columna: 25, Lexema: titulol
Número de línea: 53, Número de columna: 33, Lexema: end
Número de línea: 53, Número de columna: 36, Lexema: ;
Número de línea: 54, Número de columna: 8, Lexema: values
Número de línea: 54, Número de columna: 14, Lexema: :
Número de línea: 54, Número de columna: 15, Lexema: :
Número de línea: 54, Número de columna: 16, Lexema: double
Número de línea: 54, Número de columna: 23, Lexema: =
Número de línea: 54, Número de columna: 25, Lexema: [
Número de línea: 54, Número de columna: 26, Lexema: 2
Número de línea: 54, Número de columna: 27, Lexema: ,
Número de línea: 54, Número de columna: 29, Lexema: 2
Número de línea: 54, Número de columna: 30, Lexema: ,
Número de línea: 54, Número de columna: 32, Lexema: 2
Número de línea: 54, Número de columna: 33, Lexema: ,
Número de línea: 54, Número de columna: 35, Lexema: 5
Número de línea: 54, Número de columna: 36, Lexema: ,
```

Instrucciones de uso:

5.2 Encapsulamiento

Todas las sentencias del lenguaje deben venir entre **PROGRAM** y **END PROGRAM**

```
PROGRAM
    <CÓDIGO>
END PROGRAM
```

```
! Esto es un comentario de una sola línea
<! Esto es un comentario
Multilínea !>
```

Nota: el comentario multilínea mejor evitarlo.

5.4 Tipos de Dato

El lenguaje DataForge admite únicamente dos tipos de datos para trabajar con ellos.

Tipo	Definición	Descripción	Ejemplo
Cadena	char[ ]	Es un grupo o conjunto de caracteres que pueden tener cualquier carácter, y este se encontrará delimitado por comillas dobles. “ ”	“cadena ejemplo”
Decimal	double	Este tipo de dato admite valores numéricos	0, 10, 0.5, 57.75, etc.

5.5 Declaración de variables

Una variable debe ser declarada antes de poder ser utilizada. Todas las variables tendrán un tipo de dato y un nombre de identificador. Las variables pueden ser utilizadas escribiendo el nombre del identificador.

```
var:<TIPO>::<ID> <- <EXPRESION> end;
! Ejemplos
var:double:: numero <- 2.5 end;
var:char[]::cadena <- “cadena” end;
var:double:: copia <- numero end; ! copia tiene el valor 2.5
```

5.6.2 Declaración de Arreglos

Un arreglo debe ser declarado antes de ser utilizado. Todos los arreglos tendrán un tipo de dato y un nombre identificador que siempre inicia con el símbolo @.

```
! Declaración de arreglos
arr:<TIPO>::@<ID> <- <LISTA_VALORES> end;

! Ejemplos
arr:double::@darray <- [1, 2, 3, 4, 5] end; ! Arreglo de tipo double
arr:char[]::@carray <- [“12”, “2”, “3”] end; ! Arreglo de tipo string
arr:double::@carray <- [numero, copia, 7] end; ! Puede usar variables
```

```
! Operaciones
var:double:: suma <- SUM(5, 2) end;
var:double:: resta <- RES(3, 2) end;
var:double:: multi <- MUL(4, numero) end;  ! Funciona con variables
var:double:: division <- DIV(1, variable) end;
var:double:: modulo <- MOD(5, 4) end;

! Operaciones anidadas
var:double:: suma <- MUL( SUM(7,3) , RES(7, DIV(25,5) ) end;
arr:double::@darray <- [ SUM(7,3), DIV(25,5)] end; ! Arreglo con funciones
```

Media(<ARREGLO_DOUBLE>**)**
Mediana(<ARREGLO_DOUBLE>**)**
Moda(<ARREGLO_DOUBLE>**)**
Varianza(<ARREGLO_DOUBLE>**)**
Max(<ARREGLO_DOUBLE>**)**
Min(<ARREGLO_DOUBLE>**)**

```
! Imprime cada expresión separado por coma
console::print = <EXP>, <EXP>, ... end;
```

Ejemplo:

```
var:double:: numero <- 15 end;
console::print = "hola", numero, 15, "adios" end;
! Salida: hola, 15, adios

console::print = 1, 2, SUM(3,5), Media(@arreglo) end;
! Salida: 1, 2, 8, 15.7
```

5.9.2 Imprimir Arreglos:

Esta función permite mostrar arreglos en consola en un formato de tabla solicitando un título y un arreglo de cualquier tipo de dato. El arreglo puede ser de una variable o declarado directamente. El título puede ser ingresado directamente como una expresión o de una variable.

```
! Muestra una tabla de una columna en consola con cada línea un valor
console::column = <CADENA> -> <ARREGLO> end;
```

Ejemplo:

```
arr:double::@darray <- [1, 2, 3, 4, 5] end;
var:char[ ]:: titulo <- "Enteros" end;
console::column = "Enteros" -> @darray  end;
console::column = titulo ->  [1, 2, 3, 4, 5]  end;
```

Salida:

```
-------------
Enteros
-------------
1
2
```

5.10 Funciones de Graficación:

Para una mejor visualización de la información el lenguaje DataForge cuenta con funciones que permite graficar de manera personalizada un conjunto de datos utilizando la siguiente sintaxis:

```
<tipoGrafica> (
   <sentencias>
   EXEC <tipoGrafica> end;
) end;
```

Ejemplo:

```
graphPie(
   titulo::char[] = "Titulo inicial" end;
   label::char[] = ["dato incorrecto", "dato2" ] end;
   values::double = [20, 70] end;
   titulo::char[] = "Titulo que se debe mostrar" end;
   label::char[] = ["dato correcto", "dato2" ] end;
   EXEC grapPie end;
) end;
```

```
graphBar(
   titulo::char[] = <cadena> end;
   ejeX::char[] = <ARREGLO_CADENA> end;
   ejeY::double = <ARREGLO_DOUBLE> end;
   tituloX::char[] = <cadena> end;
   tituloY::char[] = <cadena> end;
   EXEC grapBar end;
) end;
```

```
graphLine(
    titulo::char[] = <Cadena> end;
    ejeX::char[] = <ARREGLO_CADENA> end;
    ejeY::double= <ARREGLO_DOUBLE> end;
    tituloX::char[] = <Cadena> end;
    tituloY::char[] = <Cadena> end;
    EXEC grapLine end;
) end;
```

Ejemplo:

```
Histogram(
   titulo::char[] = <Cadena> end;
   values::char[] = <ARREGLO_DOUBLE> end;
   EXEC Histogram end;
) end;
```

Ejemplo:

```
PROGRAM
   ! Variables
   var:double:: notaAprobar <- 61 end;
   var:char[]:: labelAprobar <- "Nota Minima" end;

   ! Arreglos
   arr:double:: @notas <- [notaAprobar, MUL(75, 0.45), DIV(SUM(80,20), RES(75,25))] end;
   arr:char[]:: @labels <- [labelAprobar, "P1", "P2"] end;

   ! Prints
   var:char[]:: titulo1 <- "Titulo histograma" end;
   console::column = "test" -> [10, 15.5, 61.1] end;
   console::column = "Notas" -> @notas end;
   console::column = titulo1 -> @labels end;

   console::print = "Media", "Mediana", "Moda", "Varianza", "Max", "Min" end;
   console::print = Media(@notas), Mediana(@notas), Moda(@notas), Varianza(@notas),
Max(@notas), Min(@notas) end;
   console::print = "Hola Mundo", MOD(10, 9), notaAprobar, labelAprobar end;


   var:double:: gb1 <- 61 end;
   var:char[]:: gbt <- "Datos" end;
   graphBar(
      !grafica 1
      titulo::char[] = "Estudiantes" end;
      ejeX::char[] = ["1 Parcial", "2 parcial", "Final"] end;
      ejeY::double = [gb1, 30, 70] end;
      tituloX::char[] = "Actividades" end;
      tituloY::char[] = gbt end;
      EXEC grapBar end;
   ) end;

   graphPie(


      label::char[] = ["Uno", "Dos", "Tres"] end;
      titulo::char[] = "Ejemplo Gráfica de Pie" end;
      values::double = [50, 30, 20] end;
      EXEC grapPie end;
```

```
) end;


graphLine(
!testing de variables en graficas
   titulo::char[] = "Gráfica de Línea" end;
   ejeX::char[] = @labels end;
   ejeY::double = @notas end;
   tituloX::char[] = "Actividades" end;
   tituloY::char[] = "Notas" end;
   EXEC grapLine end;
) end;

Histogram(
   titulo::char[] = titulo1 end;
   values::double = [2, 2, 2, 5, 5, 7, 8] end;
   EXEC Histogram end;
) end;



END PROGRAM
```