

José Mérida - 201105

Adrián López - 21357

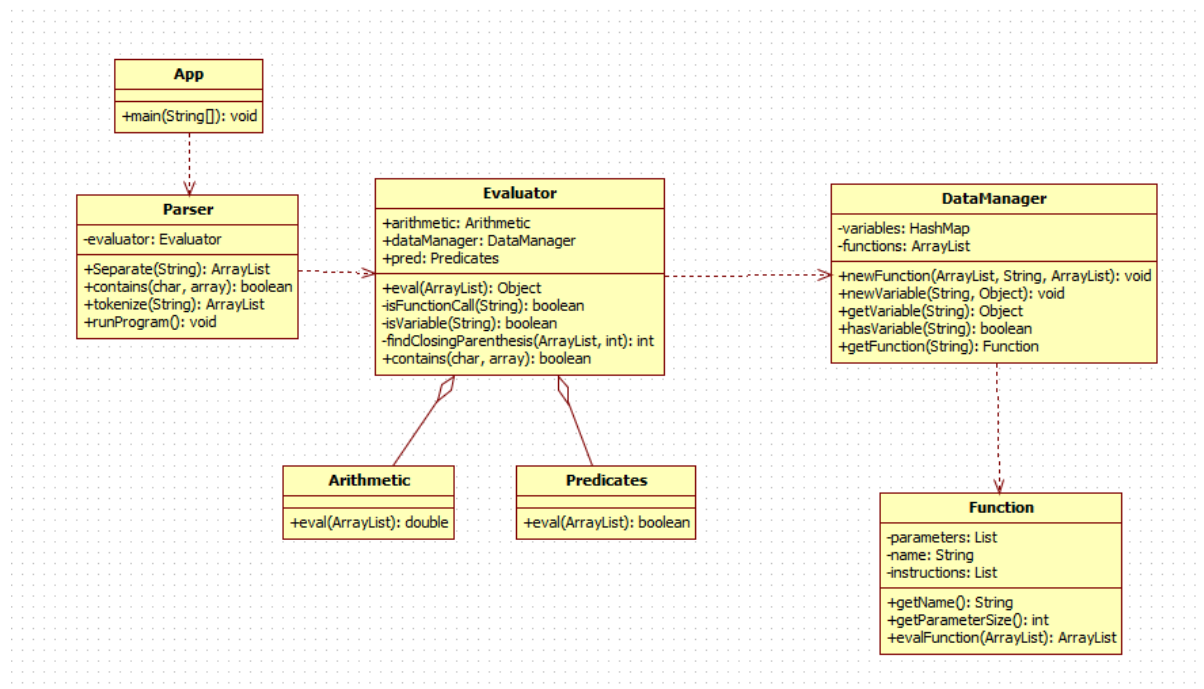
Algoritmos y Estructuras de Datos

24/03/2024

Proyecto 1 Fase 2. Intérprete LISP en Java

Diseño Final del Intérprete

Diagrama de Clases



Estructuras Utilizadas: En su mayoría trabajamos con ArrayLists, debido a su tamaño

dinámico fueron útiles para poder modificar diferentes secciones y valores por evaluar.

Únicamente utilizamos Lists dentro de la clase Function para almacenar valores que no van

a cambiar (como los parámetros o instrucciones de la función). También implementamos

HashMaps dentro de DataManager para poder almacenar variables y sus valores en pares

(key, value).

Diagrama de Estados

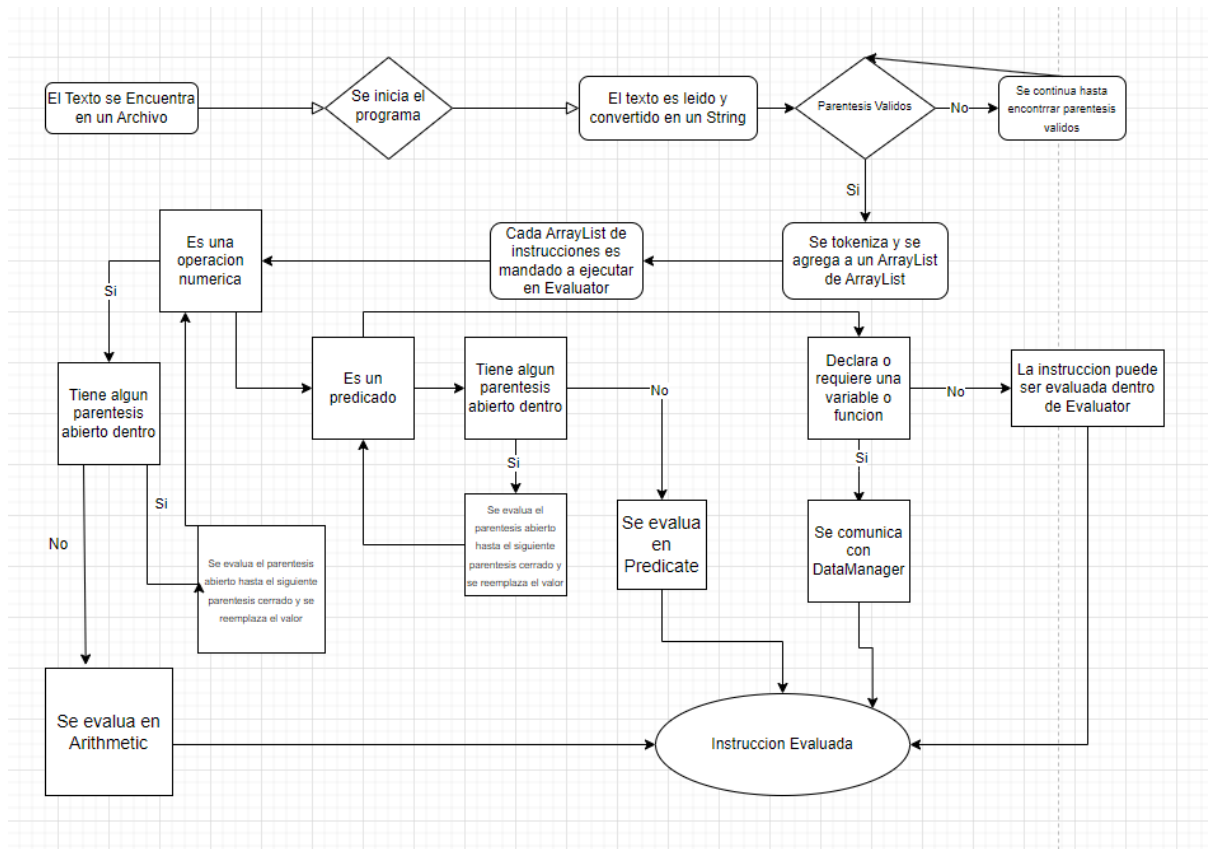
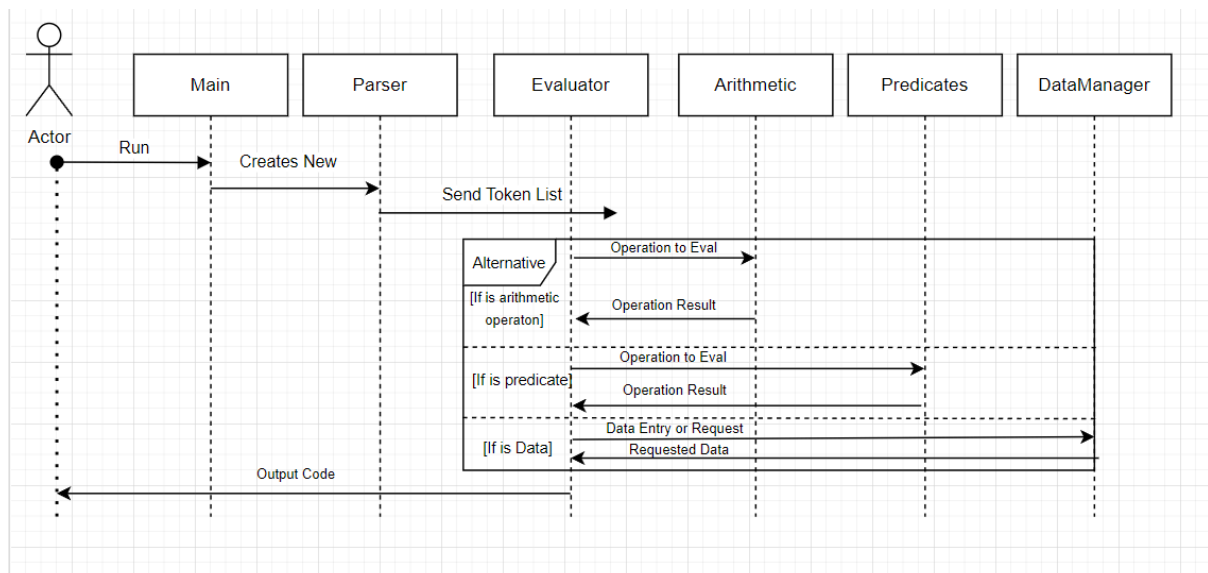


Diagrama de Secuencias

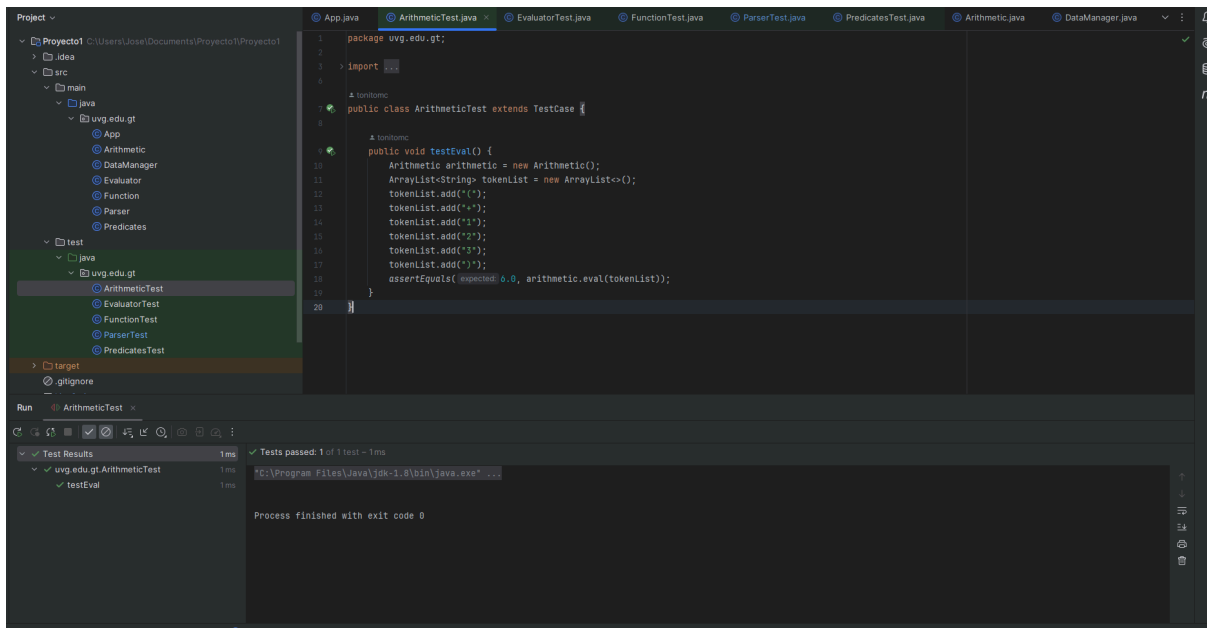


Controlador de Versiones

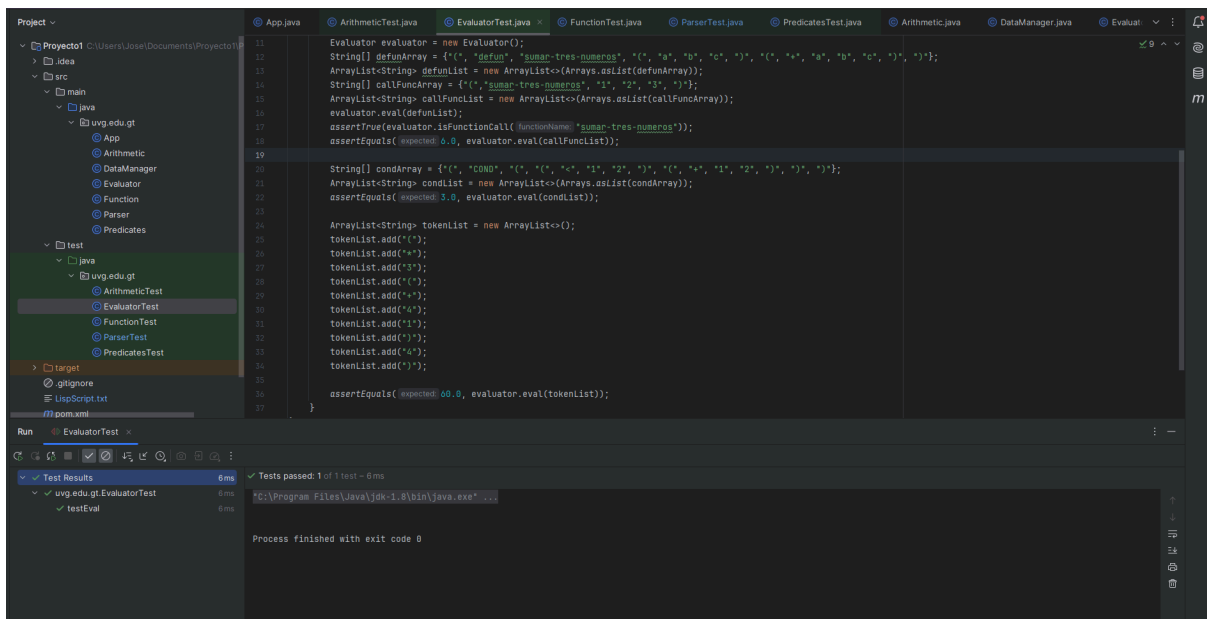
https://github.com/TonitoMC/Interprete_LISP_AED

Esquema de Pruebas Unitarias

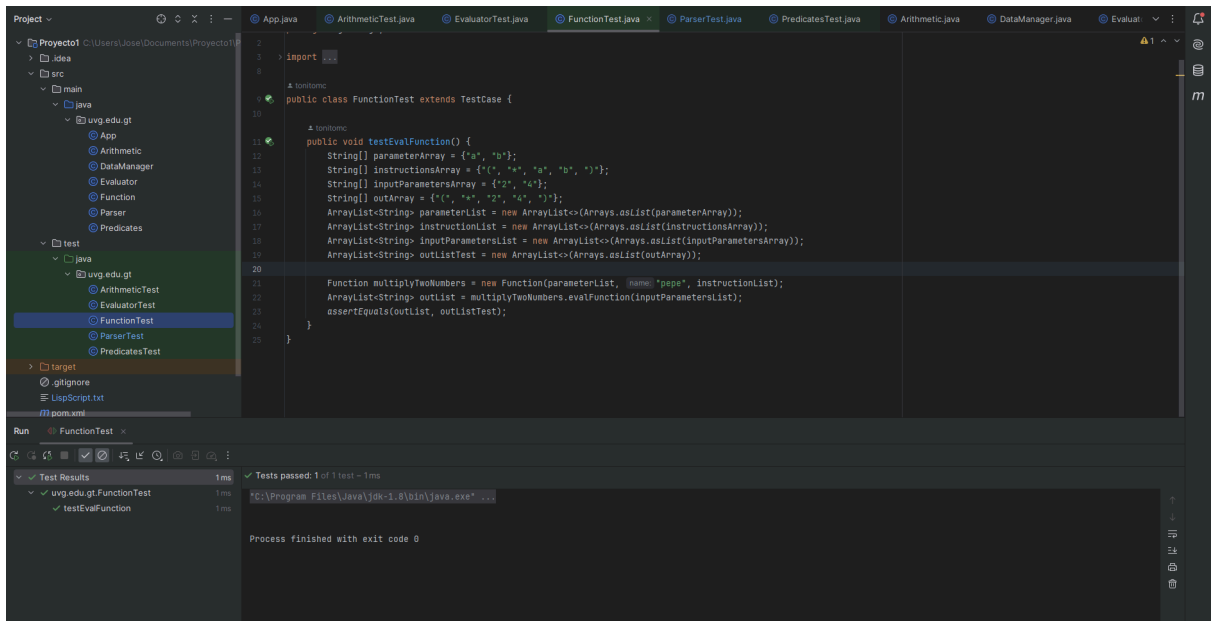
Arithmetic Test



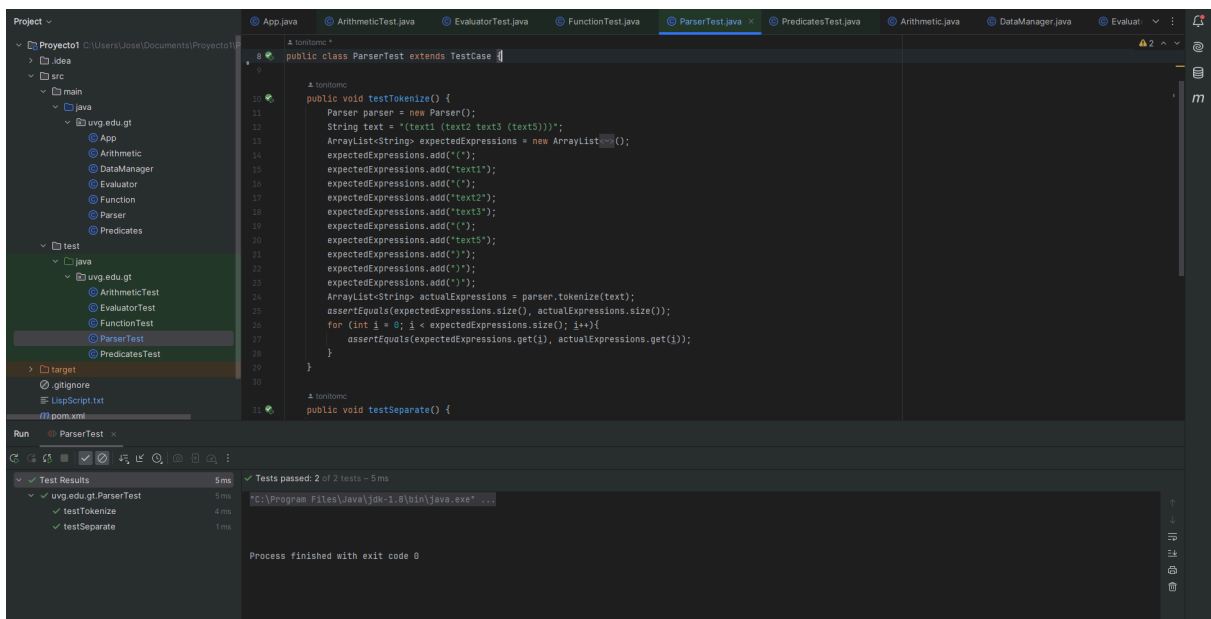
Evaluator Test



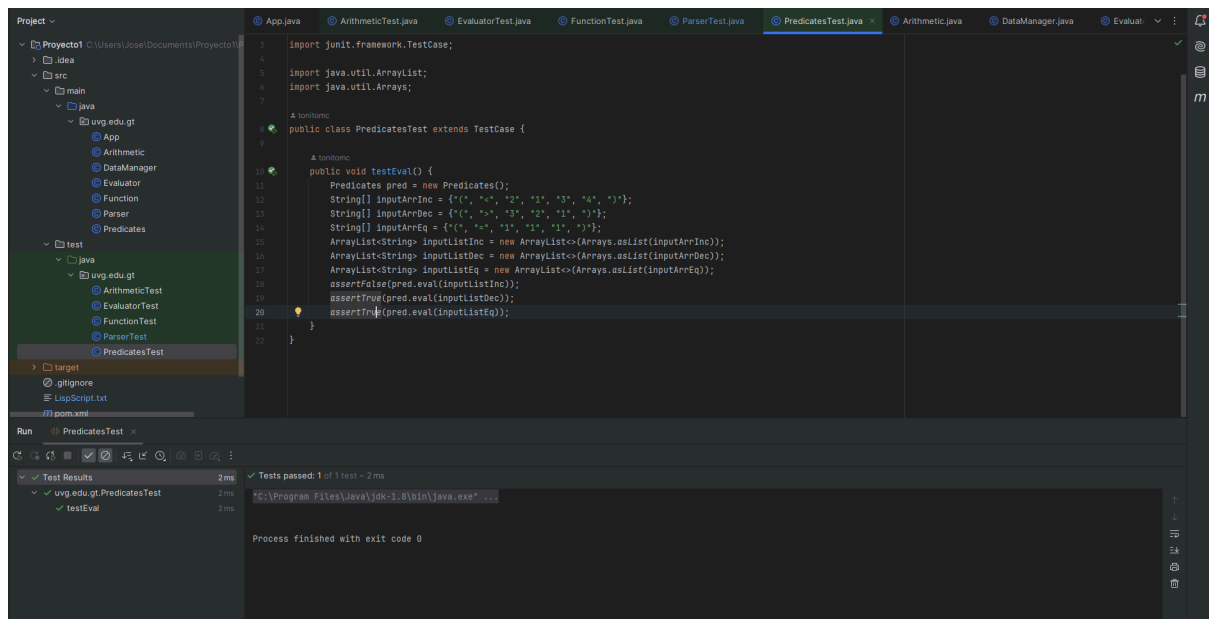
Function Test



Parser Test



Predicates Test



Debido a la dificultad de trabajar con ArrayLists en las pruebas unitarias, algunos bloques de código (no necesariamente métodos) dentro de Evaluator se manejaron directamente corriendo el programa con el archivo de texto conteniendo el código de lisp y utilizando el Debugger.

Programa por Ejecutar

El programa consta de diferentes instrucciones por ejecutar para comprobar cada una de las funcionalidades deseadas:

Operaciones Aritmeticas:

(print (+ 3 7))

(print (* 2 (- 7 4) 7))

SETQ / Imprimir Variables

(SETQ variable1 10)

(print variable1)

Condicionales, Predicados y Definición de Funciones:

```
(defun orden (num1 num2 num3)

(COND

((< num1 num2 num3) (print "Ascendente"))

(> num1 num2 num3) (print "Descendiente"))

(= num1 num2 num3) (print "Iguales"))

(t (print "Ninguno"))))

(orden 1 2 3)

(orden 3 2 1)

(orden 1 1 1)

(orden 2 1 3)
```

Atom y Listas

```
(SETQ variable8 (list 1 2 3))

(print(ATOM (variable8)))

(print(ATOM (variable1)))
```

Definición de Funciones y Paso de Parametros:

```
(defun sumartresnumeros (num1 num2 num3)

(+ num1 num2 num3))

(print (sumartresnumeros 1 2 3))

(print (sumartresnumeros (sumartresnumeros 1 2 3) 2 3))
```

Definición de Funciones y Recursividad

```
(defun factorial (n)
```

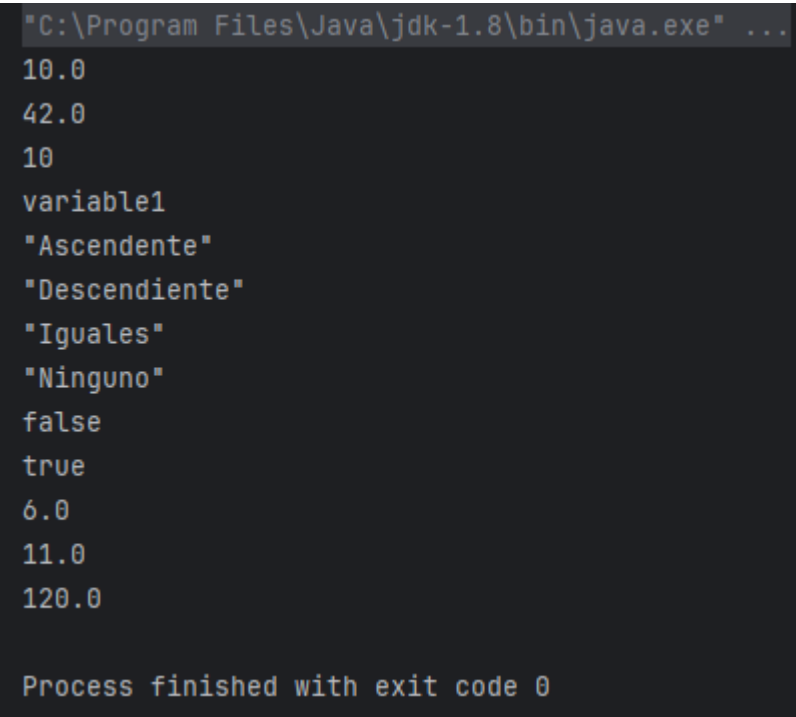
```
(COND ((= n 0) 1)
```

```
((= n 1) 1)
```

```
(t (* n (factorial (- n 1)))))
```

```
(print (factorial 5))
```

Script Completo dentro del Repositorio, output del programa:



```
"C:\Program Files\Java\jdk-1.8\bin\java.exe" ...  
10.0  
42.0  
10  
variable1  
"Ascendente"  
"Descendiente"  
"Iguales"  
"Ninguno"  
false  
true  
6.0  
11.0  
120.0  
  
Process finished with exit code 0
```

Video Demostrativo

<https://youtu.be/iEC88OQhkGQ>