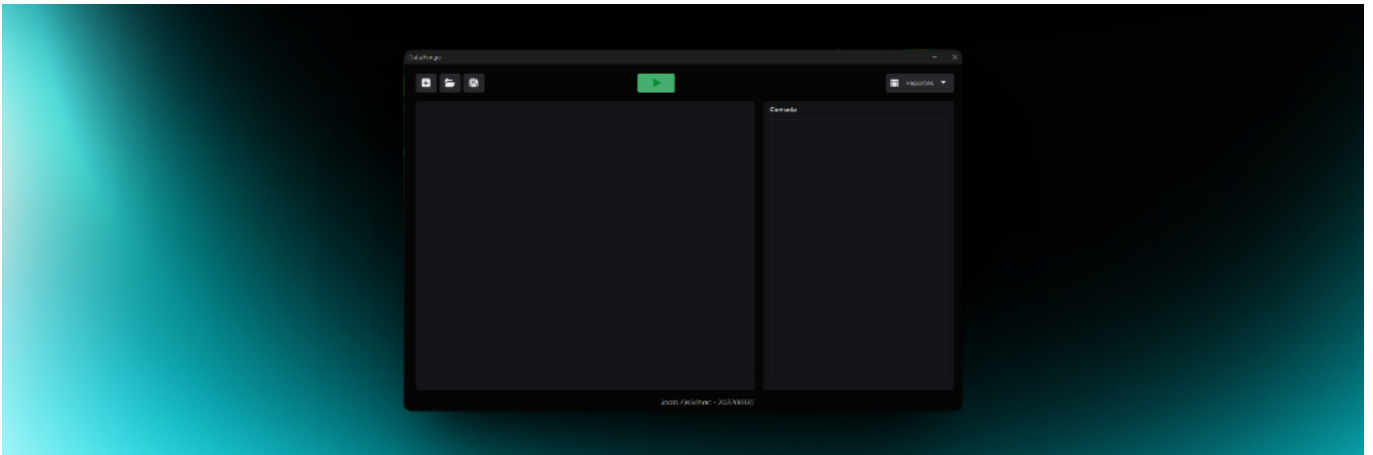







# Proyecto 1

---



 Joab Israel Ajsivinac Ajsivinac  202200135  
 Organización de Lenguajes y Compiladores 1  
 Universidad San Carlos de Guatemala  
 31 Primer Semestre 2024

## Objetivos

---

- **Objetivo General** \*Aplicar los conocimientos sobre la fase de análisis léxico y sintáctico de un compilador para la construcción de una solución de software
- **Objetivos Específicos.**
  - Aprender a genera analizadores léxicos y sintácticos utilizando JFLEX y CUP
  - aprender los conceptos de token, lexema, patrones y expresiones regulares.
  - realizar correctamente el manejo de errores léxicos.
  - realizar acciones gramaticales utilizando el lenguaje de programación JAVA.

## Manual de Usuario

---

## Requerimientos

- Sistemas Opreativos
  - Windows 8 o Superior
  - macOS Catalina o Superior
  - Linux: Ubuntu, Debian, CentOS, Fedora, etc.
- Java 19 o Superior
- NetBeans 20 o superior
- Librerías
  - FlatLaf 3.0

- flatlaf-intellij-themes-3.0
- jflex-full 1.7.0
- java-cup 11b
- java-cup 11b-runtime

## Instalación

Descargue el código o bien clone el repositorio en una carpeta.

Si se opta por la clonación se hace con la siguiente línea de código en terminal (Antes de ejecutar el código asegúrese de estar en la carpeta donde lo quiere descargar)

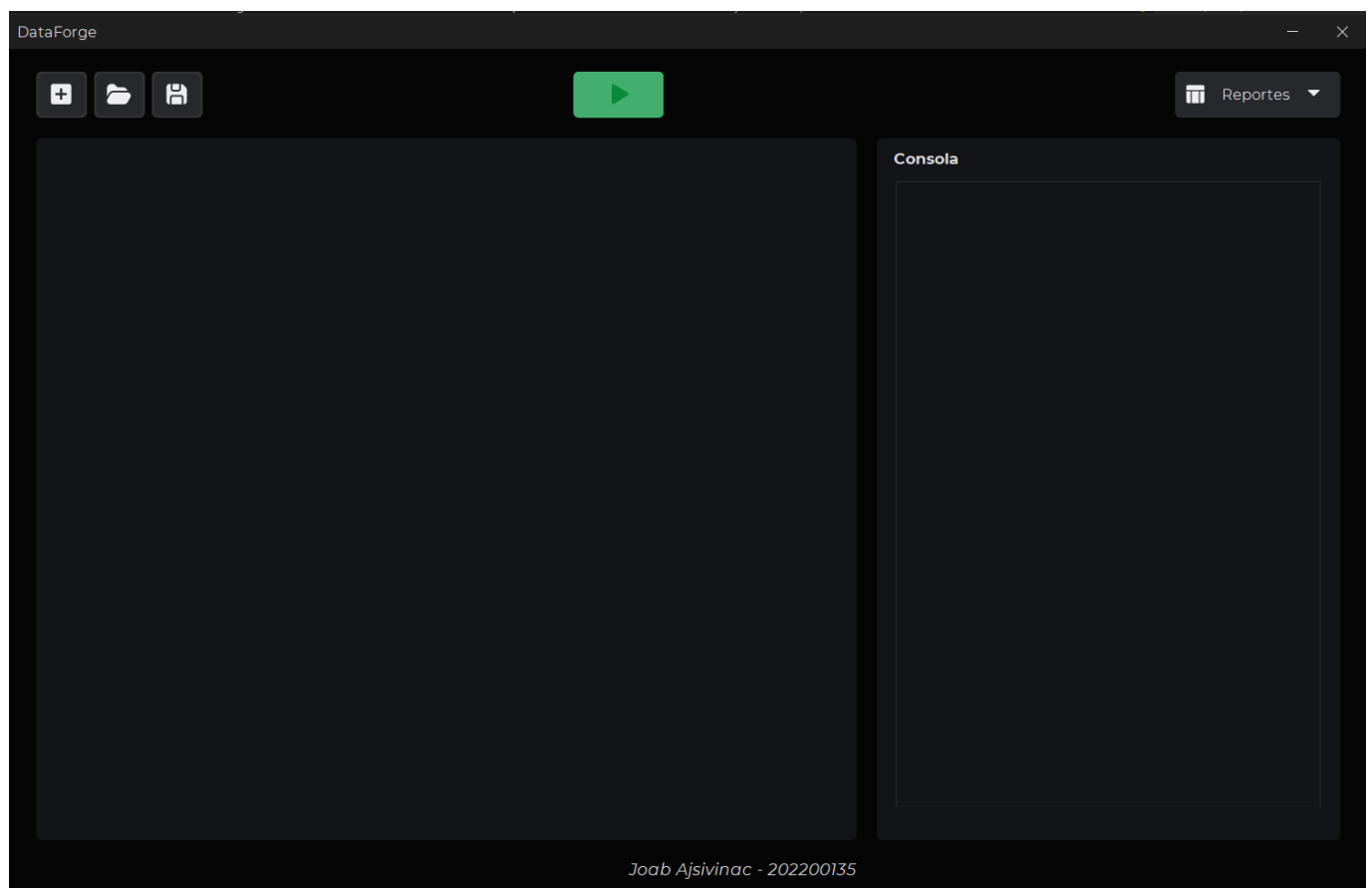
```
git clone https://github.com/J-Ajsivinac/OLC1_Proyecto1_202200135.git
```

## Inicio Rápido

Una vez teniendo el programa, abra el proyecto con Netbeans, y dirijase al paquete proyecto y abra **Proyecto1.java** y luego presione **F6** o de click para ejecutar.

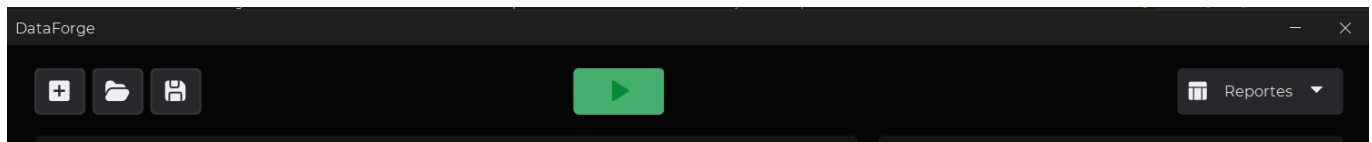
## Interfaz de Usuario y Funcionalidades

Al ejecutar el proyecto se le mostrara la siguiente ventana



El programa consta de 2 partes principales, las cuales son:

Parte superior



En esta parte se pueden realizar las siguientes acciones

**Sección Izquierda (Manejo de archivos)** En la parte izquierda podra visualizar 3 botones, los cuales están encargados de Agregar una nueva pestaña, Abrir un archivo con extensión **.df** y guardar, esto le permitira interactuar con el panel que esta debajo de esta sección

**Sección Media (Ejecución del analisis)** En la parte media, se puede visualizar un botón verde, el cual al presionarlo enviara el código escrito para su análisis

**Sección Derecha (Reportes)** En la parte derecha puede visualizar un botón el cual al presionarlo se desplegará un menú, en el cual podrá seleccionar 3 tipos de reportes:

- Reporte de Tokens: muestra los tokens reconocidos por el analizador léxico
- Reporte de Errores: muestra los errores léxicos y sintácticos encontrados durante el analisis.
- Reporte de Tabla de símbolos: muestra todas las variables y arreglos declarados

**Nota:** Todos los reportes son generados con extensión **.html**

## Parte media

En esta parte derecha se puede agregar el codigo ha analizar, y en la parte izquierda se podra visualizar las impresiones en consola generadas por el codigo escrito previamente analizado.

En la parte donde se agrega el codigo, es necesario, agregar una pestaña o bien abrir un archivo, para poder acceder al cuadro de texto donde se escribira el codigo.

Las instrucciones:

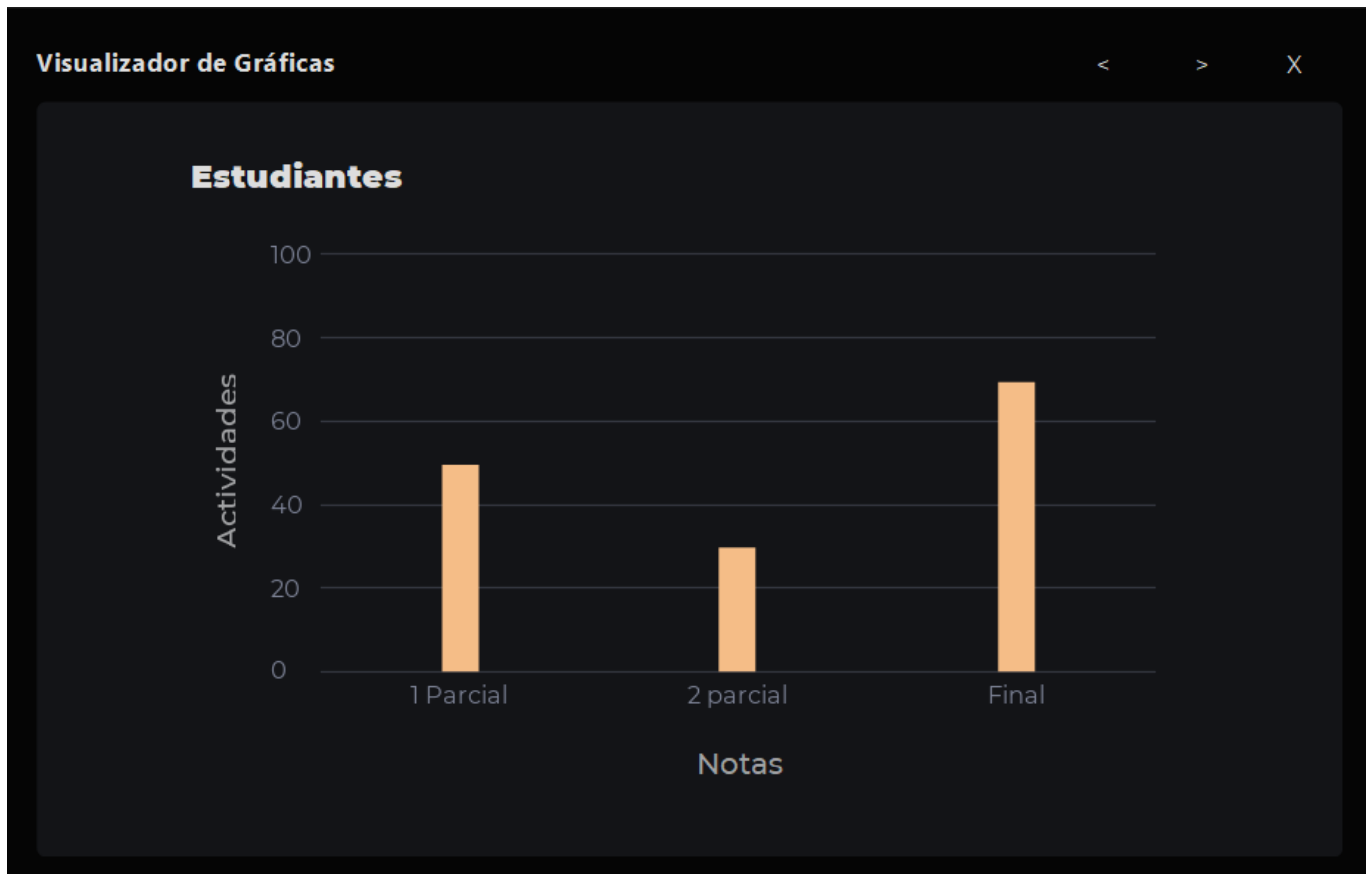
```
console::print = "hola", numero, 15, "adios" end;
console::column = "Enteros" -> @darray end;
Histogram(
  titulo::char[] = "Analisis de Arreglo" end;
  values::char[] = [2,2,2,5,5,7,8] end;
  EXEC Histogram end;
) end;
```

Muestran una salida en consola

---

El programa tambien cuenta con la opción de generar las siguientes gráficas:

- **Barras**



La gráfica de barras cuenta con la siguiente sintaxis:

```
graphBar(  
  titulo::char[] = "Estudiantes" end;  
  ejeX::char[] = ["1 Parcial", "2 parcial", "Final"] end;  
  ejeY::double = [50, 30, 70] end;  
  tituloX::char[] = "Actividades" end;  
  tituloY::char[] = "Notas" end;  
  EXEC graphBar end;  
) end;
```

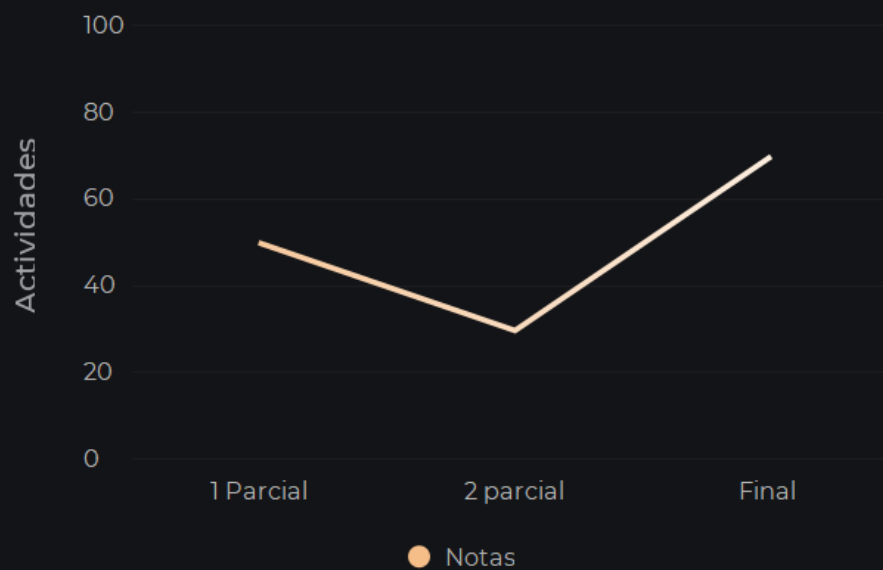
- **Pie**



La gráfica de Pie cuenta con la siguiente sintaxis:

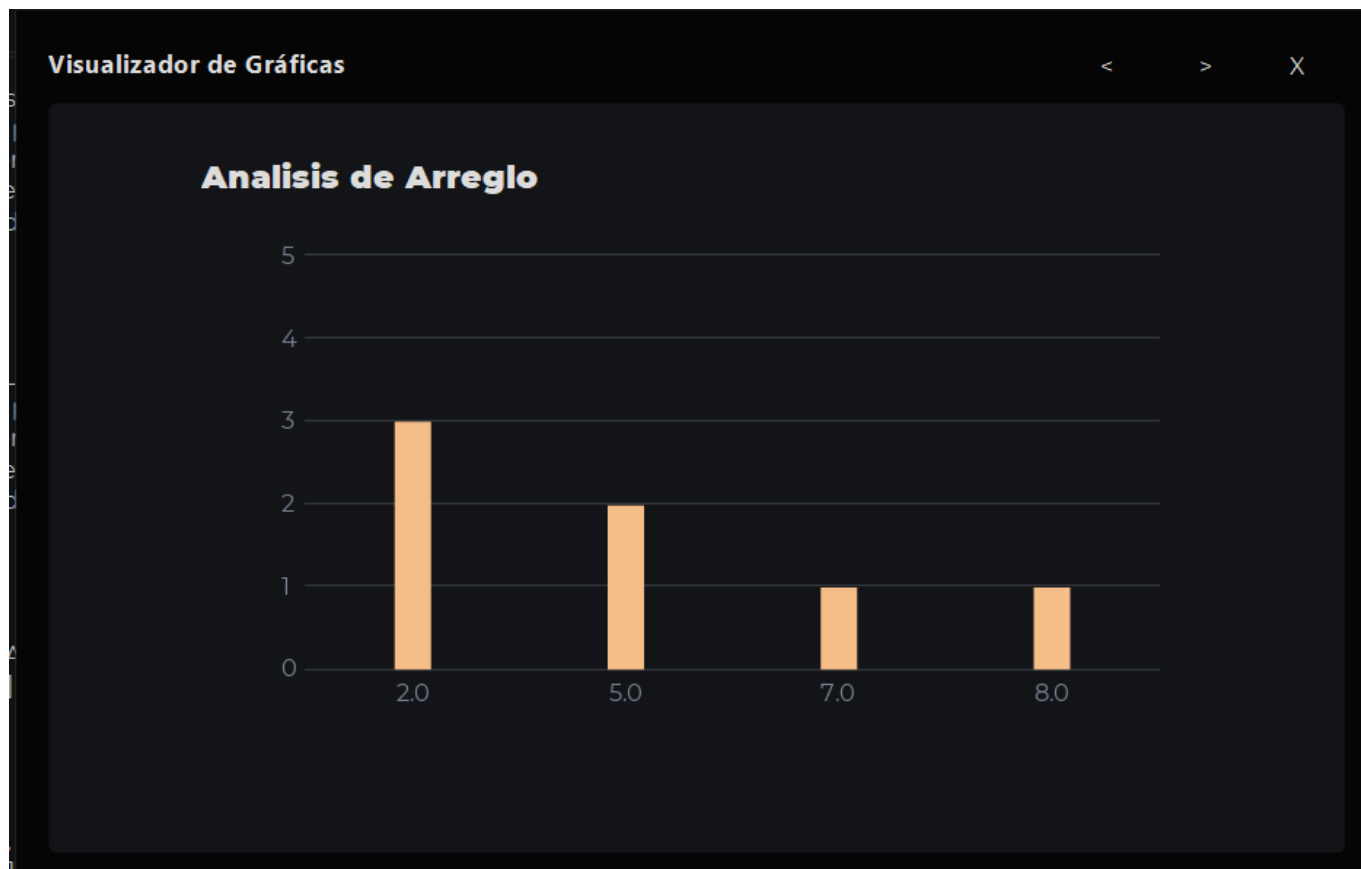
```
graphLine(  
  titulo::char[] = "Gráfica de Línea" end;  
  ejeX::char[] = ["1 Parcial", "2 parcial", "Final"] end;  
  ejeY::double = [50, 30, 70] end;  
  tituloX::char[] = "Actividades" end;  
  tituloY::char[] = "Notas" end;  
  EXEC graphLine end;  
) end;
```

- **Línea**

**Gráfica de Línea**

```
graphLine(  
  titulo::char[] = "Gráfica de Línea" end;  
  ejeX::char[] = ["1 Parcial", "2 parcial", "Final"] end;  
  ejeY::double = [50, 30, 70] end;  
  tituloX::char[] = "Actividades" end;  
  tituloY::char[] = "Notas" end;  
  EXEC graphLine end;  
) end;
```

- **Histograma**



La gráfica de barras cuenta con la siguiente sintaxis:

```
Histogram(  
    titulo::char[] = "Analisis de Arreglo" end;  
    values::char[] = [2,2,2,5,5,7,8] end;  
    EXEC Histogram end;  
) end;
```

## Reportes

Como se indico anteriormente el programa cuenta con la capacidad de realizar reportes en formato **.html**, los cuales se guardan en la carpeta raiz del proyecto, en una carpeta llamada **Reports**, ahi encontrará los archivos con sus respectivos nombres.

**reporte\_tokens.html**

## Titulo del Reporte

No	Lexema	Tipo	Línea	Columna
1	program	TK_program	1	1
2	graphBar	TK_graphbar	2	4
3	(	TK_lparen	2	12
4	titulo	TK_titulo	3	8
5	:	TK_colon	3	14
6	:	TK_colon	3	15
7	char	TK_char	3	16
8	[	TK_lbracket	3	20

## reporte\_errores.html

### Reporte de Errores

#### Léxicos

No	Descripción	Línea	Columna
1	El caracter : '@' No pertenece al lenguaje	2	2
2	El caracter : '?' No pertenece al lenguaje	2	4

#### Sintácticos

No	Descripción	Línea	Columna
1	El caracter : 'end' no se esperaba	0	34

## reporte\_tabla.html



## Tabla de Símbolos

No	Descripción	Línea	Columna
1	numero	variable Double	2.5
2	cadena	variable String	"cadena"
3	copia	variable Double	2.5
4	@darray	arreglo	[1.0,2.0,3.0,4.0,5.0]
5	@carray	arreglo	[2.5,2.5,7.0]
6	suma	variable String	7.0
7	resta	variable String	1.0
8	multi	variable String	10.0