

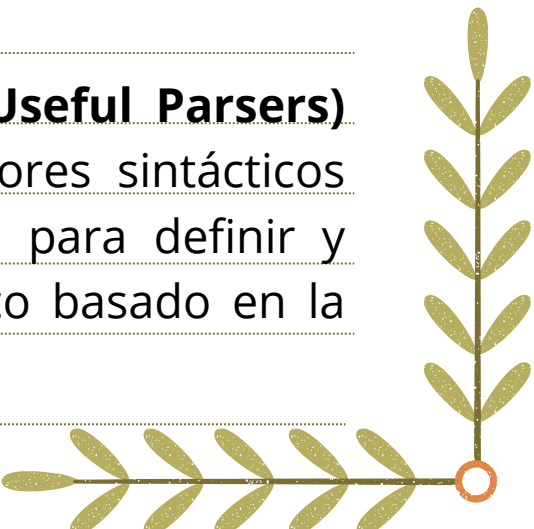


# Manual Técnico

Un analizador léxico convierte el texto de entrada en una secuencia de tokens. Los tokens son unidades léxicas como identificadores, palabras clave, números y símbolos. El analizador sintáctico analiza la estructura gramatical de la secuencia de tokens generada por el analizador léxico. Utiliza una gramática formal para verificar si la secuencia de tokens sigue las reglas sintácticas del lenguaje de programación.

Para mejorar este proyecto en un futuro se podrían agregar mas funcionalidades ya que esta es muy limitada para eso es necesario tener conocimiento de las herramientas utilizadas.

## Herramientas:

1. **JFlex:** **JFlex** es una herramienta para generar analizadores léxicos (scanners) en Java. Se utiliza para analizar el texto de entrada y dividirlo en tokens para el análisis sintáctico posterior.
  2. **CUP: CUP (Construction of Useful Parsers)** es un generador de analizadores sintácticos (parsers) para Java. Se utiliza para definir y generar el analizador sintáctico basado en la gramática especificada.
- 



**3. JFreeChart:** es una biblioteca Java que permite la creación de gráficos, incluidos gráficos de barras, líneas, pasteles, etc. En tu proyecto, parece que se utilizará para generar diversos gráficos a partir de los datos analizados.

### **Estructura del código**

La aplicación esta contenida en un 5 paquetes donde se desarrollan las distintas clases y formularios necesarios para la ejecución correcta de la aplicación el paquete contiene 1 JFrame Form y 13 Java class 1 jflex y 1 cup.

### **Explicación y mención de clases relevantes:**

Entre las distintas clases que se desarrollan tenemos una 3 archivos importantes el jflex que contiene el analizador léxico, el cup que es el que se encarga de generar el analizador sintáctico, y luego tenemos nuestra clase Interprete que es la que recibe el árbol semántico del cup y va linea por linea interpretándola.



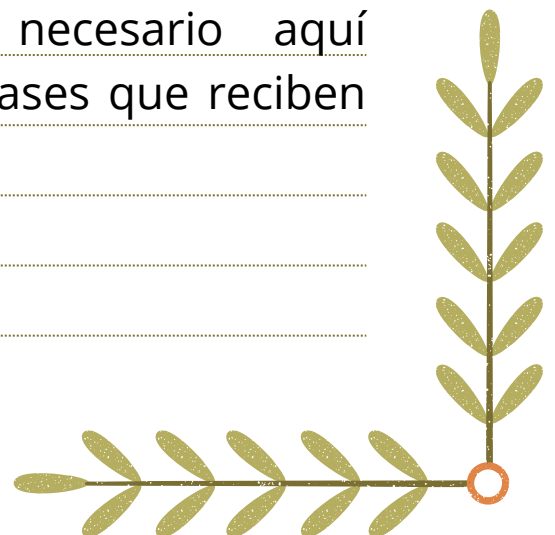



Tenemos una clase especial para los reportes y las graficas los reportes son archivos html que se generan y al darle clic al botón correspondiente se abrirán.

La clase de graficas recibe todos los parámetros de las graficas necesarios para cada uno y tiene las 4 funciones para cada uno de las graficas.

Optimización: El código de este programa aun se puede optimizar bastante se trabajo el interprete por medio de un árbol que se fue leyendo y eliminando de poco a poco para poder generar las instrucciones muchas de sus partes son repetitivas se pueden almacenar en funciones y métodos para la optimización de código.

Existen distintas clases de objetos como token, Valores, Instrucción, Error\_ y Símbolo las cuales registran cada uno de los tokens que son importantes para cada cosa necesario aquí podemos optimizar ya que hay clases que reciben lo mismo.






```
while (!this.instrucciones.isEmpty()) {
    String instru = this.instrucciones.remove(0).getLexema();
    System.out.println(instru);
    switch (instru) {
        case ("var"):
            this.instrucciones.remove(0); //:
            this.declaracionVariable();
            break;
        case ("arr") :
            this.instrucciones.remove(0); //:
            this.declaracionarreglo();
            break;
        case ("CONSOLE"):
            this.instrucciones.remove(0); //:
            this.imprimir();
            break;

        case ("BAR"): //bar y line tienen los mismo parametros
            this.instrucciones.remove(0); //(
            this.graficarbaryline("BAR");
            break;
        case ("LINE"):
            this.instrucciones.remove(0); //(
            this.graficarbaryline("LINE");
            break;
        case ("PIE"):
            this.instrucciones.remove(0); //(
            this.graficarbaryline("PIE");
            break;

        case ("HISTOGRAM"):
            this.instrucciones.remove(0); //(
            this.graficarbaryline("HISTOGRAM");
            break;
    }
}
```

Este código es importante por que es donde comienza la interpretación del lenguaje ya que estas palabras son las que nos indicaran una accion ya sea una grafica, una declaración, una impresión y otras cosas.



# Diagrama de clases

```
public void declaracionVariable() {  
    String tipo = this.instrucciones.remove(0).getLexema(); //obtiene el tipo de dato  
    this.instrucciones.remove(0); // ::  
    Instruccion variable = this.instrucciones.remove(0); //obtiene el id  
    this.instrucciones.remove(0); //<-  
    switch (tipo) {  
        case ("double"):  
            float numero = this.getvalorfloat();  
            this.hash.put(variable.getLexema(), new Valores(variable.getLinea(),  
                variable.getColumna(), variable.getLexema(),  
                null, numero, null, null, "variable double"));  
            break;  
        case ("char[]"):  
            String valor = this.instrucciones.remove(0).getLexema();  
            //AQUI O ES STRIND O ID  
            if (this.hash.containsKey(valor)){  
                valor = this.hash.get(valor).getValorC();  
            }  
            this.hash.put(variable.getLexema(), new Valores(variable.getLinea(),  
                variable.getColumna(), variable.getLexema(), valor,  
                (float) 4543333333433.333, null, null, "variable char"));  
            break;  
    }  
    this.instrucciones.remove(0); //end  
    this.instrucciones.remove(0); //;  
}
```

Este es una parte de como se interpreto la declaracion de variables y arreglos para la lectura del archivo el cual es un codigo bastante repetitivo ya que servira en otras llamadas.