Manual Técnico

Introducción:

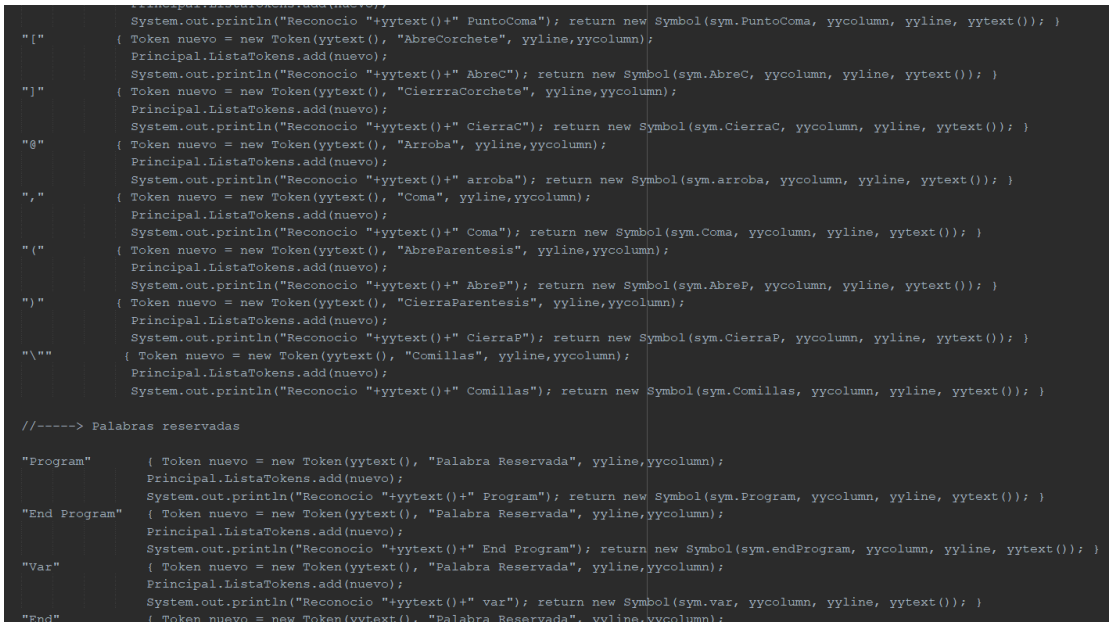
La aplicación, el cual se construyó con un analizador Léxico, y un analizador sintáctico. Nos permite la lectura de

Datos del sistema donde se realizó:

- Windows 10 Home
- Intel(R) Core(TM) i7-8565U CPU @ 180GHz 1.99 GHz
- 12 GB de memoria RAM.
- Sistema operativo de 64 bits.
- java versión "1.8.0_281"
- NetBeans IDE 8.1
- JFlex 1.6.1
- JCup 11b

Diccionario de Clases;

- A_Lexico.jflex: Dentro de esta clase que se utiliza JFlex, se realiza toda laconstrucción léxica con la cual nuestro programa va a poder aceptar los caracteres.

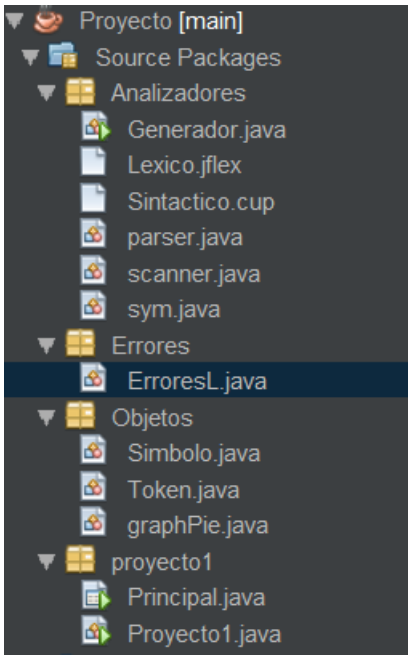


```
                System.out.println("Reconocio "+yytext()+" PuntoComa"); return new Symbol(sym.PuntoComa, yycolumn, yyline, yytext()); }
"["           { Token nuevo = new Token(yytext(), "AbreCorchete", yyline,yycolumn);
                Principal.ListaTokens.add(nuevo);
                System.out.println("Reconocio "+yytext()+" AbreC"); return new Symbol(sym.AbreC, yycolumn, yyline, yytext()); }
"]"           { Token nuevo = new Token(yytext(), "CierrraCorchete", yyline,yycolumn);
                Principal.ListaTokens.add(nuevo);
                System.out.println("Reconocio "+yytext()+" CierraC"); return new Symbol(sym.CierraC, yycolumn, yyline, yytext()); }
"@"           { Token nuevo = new Token(yytext(), "Arroba", yyline,yycolumn);
                Principal.ListaTokens.add(nuevo);
                System.out.println("Reconocio "+yytext()+" arroba"); return new Symbol(sym.arroba, yycolumn, yyline, yytext()); }
","           { Token nuevo = new Token(yytext(), "Coma", yyline,yycolumn);
                Principal.ListaTokens.add(nuevo);
                System.out.println("Reconocio "+yytext()+" Coma"); return new Symbol(sym.Coma, yycolumn, yyline, yytext()); }
"("           { Token nuevo = new Token(yytext(), "AbreParentesis", yyline,yycolumn);
                Principal.ListaTokens.add(nuevo);
                System.out.println("Reconocio "+yytext()+" AbreP"); return new Symbol(sym.AbreP, yycolumn, yyline, yytext()); }
")"           { Token nuevo = new Token(yytext(), "CierraParentesis", yyline,yycolumn);
                Principal.ListaTokens.add(nuevo);
                System.out.println("Reconocio "+yytext()+" CierraP"); return new Symbol(sym.CierraP, yycolumn, yyline, yytext()); }
"\""          { Token nuevo = new Token(yytext(), "Comillas", yyline,yycolumn);
                Principal.ListaTokens.add(nuevo);
                System.out.println("Reconocio "+yytext()+" Comillas"); return new Symbol(sym.Comillas, yycolumn, yyline, yytext()); }

//-----> Palabras reservadas

"Program"       { Token nuevo = new Token(yytext(), "Palabra Reservada", yyline,yycolumn);
                  Principal.ListaTokens.add(nuevo);
                  System.out.println("Reconocio "+yytext()+" Program"); return new Symbol(sym.Program, yycolumn, yyline, yytext()); }
"End Program"   { Token nuevo = new Token(yytext(), "Palabra Reservada", yyline,yycolumn);
                  Principal.ListaTokens.add(nuevo);
                  System.out.println("Reconocio "+yytext()+" End Program"); return new Symbol(sym.endProgram, yycolumn, yyline, yytext()); }
"Var"           { Token nuevo = new Token(yytext(), "Palabra Reservada", yyline,yycolumn);
                  Principal.ListaTokens.add(nuevo);
                  System.out.println("Reconocio "+yytext()+" var"); return new Symbol(sym.var, yycolumn, yyline, yytext()); }
"End"           { Token nuevo = new Token(yytext(), "Palabra Reservada", yyline,yycolumn);
```

- A_Sintactico.cup : Dentro de esta clase que se utiliza cup, se escriben todas las reglas semánticas las cuales nuestro programa podrá aceptar, del mismo también leemos los caracteres necesarios para que luego se puedan construir con lenguaje java los métodos.

```
terminal DosPuntos, menor, mayor, guion, PuntoComa, AbreC, CierraC, arroba, Coma, AbreP, CierraP, Igual, Comillas;
terminal Program, endProgram, var, end, doubleee, charr, arr;
terminal SUM, RES, MUL, DIV, MOD, Media, Mediana, Moda, varianza, max, min, Console, print;
terminal column, exec, graphBar, graphPie, graphLine, Histogram, values, label, titulo, ejeX, ejeY, tituloX, tituloY;
terminal cadena, doublee, id;

nonterminal String INICIO;
nonterminal Imprimir, Operaciones;
nonterminal DeclaracionDou, DeclaracionCha, DeclaracionAr, TipoAr, TipoArD, TipoArC;
nonterminal ImpresionP, TipoImpresionP;
nonterminal GraphPie;
nonterminal TodoTipo;


start with INICIO;

INICIO ::= Program TodoTipo endProgram
{:RESULT = codigoUsr; :}
;

TodoTipo ::= DeclaracionDou TodoTipo
           | DeclaracionCha TodoTipo
           | DeclaracionAr TodoTipo
           | ImpresionP
           | GraphPie
;

DeclaracionDou ::= var DosPuntos doubleee DosPuntos DosPuntos id menor guion doublee:e end PuntoComa
{: System.out.println(e);:}
;

DeclaracionCha ::= var DosPuntos charr AbreC CierraC DosPuntos DosPuntos id menor guion cadena:c end PuntoComa
{: System.out.println(c);:}
;
```

- El proyecto esta estructurado por las carpetas Analizadores, Errores, Objetos y la carpeta principal donde se encuentra el método main del proyecto y la interfaz gráfica del mismo.



- Errores: Dentro de este se almacenan los errores que puedan surgir por medio de nuestros analizadores, un orden o símbolo no aceptado.

```
public class ErroresL {
    String tipo;
    String caracter;
    int fila;
    int columna;

    public ErroresL(String tipo, String caracter,int fila,int columna) {
        this.tipo = tipo;
        this.caracter = caracter;
        this.fila = fila;
        this.columna = columna;
    }
```