Analizadores de léxico

Proyecto sobre analizadores de léxico y sintaxis en java mediante las librerias de Jcup y Jflex

Autores

- Miguel Ángel Rico García - **20191020107**
- Sergio David Paez Suarez - **20191020167**

Cambios en los archivos

Los principales cambios que se realizaron sobre el proyecto original de UDIN fueron:

```
Modificación del regex para el manejo de números sean negativos,decimales
o que cumpla ambas condicones
```

```
("(-"{D}+")")|({D}+"."+{D})|("(-"{D}+"."+{D}+")")|{D}+|("-"+{D}+"."+{D})|("-"+{D}) {lexemas=
```

```
Separación de la agrupación para los tipos de datos que se manejen
de las variables
```

```
/* Tipos de datos */
(byte) {lexemas=yytext(); return Byte;}
(int)  {lexemas=yytext(); return Int;}
(char) {lexemas=yytext(); return Char;}
(short) {lexemas=yytext(); return Short;}
(long) {lexemas=yytext(); return Long}
(float) {lexemas=yytext(); return Float;}
(double) {lexemas=yytext(); return Double;}
(String) {lexemas=yytext(); return Cadena;}
```

```
Se agregarón más tokens para el manejo del analizador lexico,
principalmente los expuestos en la plataforma.
```

```
Tipos de dato nativos */
(#include) {lexemas=yytext(); return Include}
(define) {lexemas=yytext(); return Define;}
(break) {lexemas=yytext(); return Break;}
(const) {lexemas=yytext(); return Const;}
(continue) {lexemas=yytext(); return Continue;}
(default) {lexemas=yytext(); return Default;}
(register) {lexemas=yytext(); return Register;}
(unsigned) {lexemas=yytext(); return Unsigned;}
(struct) {lexemas=yytext(); return Struct;}
(switch) {lexemas=yytext(); return Switch;}
(case) {lexemas=yytext(); return Case;}
(typedef) {lexemas=yytext(); return Typedef;}
```

```
/* Tipos de dato funciones*/
( printf ) {lexemas=yytext(); return Printf;}
( scanf ) {lexemas=yytext(); return Scanf;}
( cin ) {lexemas=yytext(); return Cin;}
( cout ) {lexemas=yytext(); return Cout;}
( using ) {lexemas=yytext(); return Using;}
( namespace ) {lexemas=yytext(); return Namespace;}
( std ) {lexemas=yytext(); return Std;}
( void ) {lexemas=yytext(); return Void;}
```

Tambien se Hizo la separación de los Operadores Logicos,Númericos y de Asignación

```
/* Operador Igual */
("=") {lexemas=yytext(); return Asignation;}
/* Operador Suma */
("+") {lexemas=yytext(); return Add;}
/* Operador Resta */
("-") {lexemas=yytext(); return Resta;}
/* Operador Multiplicacion */
("*") {lexemas=yytext(); return Times;}
/* Operador Division */
("/") {lexemas=yytext(); return Divide;}
/* Operador Modulo */
("%") {lexemas=yytext(); return Module;}

/* Operadores logicos */
(true) {lexemas=yytext(); return True;}
(false) {lexemas=yytext(); return False;}
("&&") {lexemas=yytext(); return DoubleAnd;}
("||") {lexemas=yytext(); return DoubleOr;}
("!") {lexemas=yytext(); return Not;}
("&") {lexemas=yytext(); return And;}
("|") {lexemas=yytext(); return Or;}

/*Operadores Relacionales */
(">") {lexemas = yytext(); return GreatherThan;}
("<") {lexemas = yytext(); return LessThan;}
("==") {lexemas = yytext(); return Equal;}
("!=") {lexemas = yytext(); return NotEqual;}
(">=") {lexemas = yytext(); return GreaterEqualThan;}
("<=") {lexemas = yytext(); return LessEqualThan;}
("<<") {lexemas = yytext(); return LeftShift;}
(">>") {lexemas = yytext(); return RightShift;}

/* Operadores Atribucion */
```

```
("+=") {lexemas = yytext(); return PlusEqual;}
("++") {lexemas = yytext(); return Increment;}
("--") {lexemas = yytext(); return Decrement;}
("-=") {lexemas = yytext(); return MinusEqual;}
("*=") {lexemas = yytext(); return TimesEqual;}
("/=") {lexemas = yytext(); return DivideEqual;}
("%=") {lexemas = yytext(); return ModuleEqual;}
```