//\* --------------------------Seccion codigo-usuario ------------------------\*/

import java.io.\*;

%%

/\* -----------------Seccion de opciones y declaraciones -----------------\*/

/\*--OPCIONES --\*/

/\* Nombre de la clase generada para el analizadorlexico\*/

%class analex

/\* Indicar funcionamiento autonomo

%standalone \*/

/\* Posibilitar acceso a la columna y fila actual de analisis\*/

%line

%column

/\* Habilitar la compatibilidad con el interfaz CUP para el generador sintáctico

%cup \*/

/\*--DECLARACIONES --\*/

%{

/\* Creamos una instancia del analizador léxico y lo ejecutamos sobre el argumento dado en la entrada (sacamos un resumen de los caracteres, palabras) \*/

//these variables are defined static because they will be referenced in main, a static method

static int numCount =0, identCount = 0, lineCount = 0;

public static void main(String [] args) throws IOException

{

//create a scanner and use the scanner's yylex function

//if you want standard input, System.in instead of new FileReader(args[0])

analex lexer = new analex(new FileReader(args[0]));

lexer.yylex();

System.out.println("Numeros: " + numCount + " Identificadores: " + identCount +" Lines: " +lineCount);

}

%} /\* Fin Declaraciones \*/

%type Object

/\* Declaraciones de macros NL(nueva linea) BLANCO(espacio en blanco) y TAB(tabulador) \*/

NL = \r | \r\n | \n

BLANCO = " "

TAB = \t

%state STRING, IN\_COMMENT

%%

/\* ------------------------Sección de reglas y acciones ----------------------\*/

<YYINITIAL> {

“#”.\* {/\*SL\_COMMENT - DO NOTHING\*/}

“=begin ” { yybegin(IN\_COMMENT); }

"require" {System.out.println("Token require <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"nil" {System.out.println("Token nil <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"when" {System.out.println("Token when <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"def" {System.out.println("Token def <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"false" {System.out.println("Token false <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"not" {System.out.println("Token not <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"while" {System.out.println("Token while <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"for" {System.out.println("Token for <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"then" {System.out.println("Token then <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"do" {System.out.println("Token do <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"if" {System.out.println("Token if <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"true" {System.out.println("Token true <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"begin" {System.out.println("Token begin <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"else" {System.out.println("Token else <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"break" {System.out.println("Token break <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"elsif" {System.out.println("Token elsif <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"retry" {System.out.println("Token retry <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"unless" {System.out.println("Token unless <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"case" {System.out.println("Token case <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"end" {System.out.println("Token end <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"return" {System.out.println("Token return <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"until" {System.out.println("Token until <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"," {System.out.println("Token comma <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

";" {System.out.println("Token semicolon <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"&" {System.out.println("Token Bit\_And <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"|" {System.out.println("Token Bit\_Or <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"^" {System.out.println("Token Bit\_Xor <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"~" {System.out.println("Token Bit\_Not <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"<<" {System.out.println("Token Bit\_Shl <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

">>" {System.out.println("Token Bit\_Shr <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"+=" {System.out.println("Token Plus\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"-=" {System.out.println("Token Minus\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"+" {System.out.println("Token Plus <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"-" {System.out.println("Token Minus <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"/" {System.out.println("Token Div <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"\*\*" {System.out.println("Token Exp <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"\*" {System.out.println("Token Mul <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"+" {System.out.println("Token Plus <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"%" {System.out.println("Token Mod <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"==" {System.out.println("Token Equal <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"!=" {System.out.println("Token Not\_Equal <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"<=" {System.out.println("Token Less\_Equal <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

">=" {System.out.println("Token Greater\_Equal <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"<" {System.out.println("Token Less <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

">" {System.out.println("Token Greater <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"=" {System.out.println("Token Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"\*\*=" {System.out.println("Token Exp\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"\*=" {System.out.println("Token Mul\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"/=" {System.out.println("Token Div\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"%=" {System.out.println("Token Mod\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"//\"" {System.out.println("Token Escaped\_Quote <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

"&" {System.out.println("Token Bit\_And <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"|" {System.out.println("Token Bit\_Or <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"^" {System.out.println("Token Bit\_Xor <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"~" {System.out.println("Token Bit\_Not <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"<<" {System.out.println("Token Bit\_Shl <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

">>" {System.out.println("Token Bit\_Shr <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"+=" {System.out.println("Token Plus\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"-=" {System.out.println("Token Minus\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"+" {System.out.println("Token Plus <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"-" {System.out.println("Token Minus <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"/" {System.out.println("Token Div <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"\*\*" {System.out.println("Token Exp <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"\*" {System.out.println("Token Mul <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"%" {System.out.println("Token Mod <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"==" {System.out.println("Token Equal <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"!=" {System.out.println("Token Not\_Equal <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"<=" {System.out.println("Token Less\_Equal <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

">=" {System.out.println("Token Greater\_Equal <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"<" {System.out.println("Token Less <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

">" {System.out.println("Token Greater <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"=" {System.out.println("Token Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"\*\*=" {System.out.println("Token Exp\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"\*=" {System.out.println("Token Mul\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"/=" {System.out.println("Token Div\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"%=" {System.out.println("Token Mod\_Assign <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"and" | "&&" {System.out.println("Token AND <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

"or" | "||" {System.out.println("Token OR <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

"(" {System.out.println("Token ( <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

")" {System.out.println("Token ) <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

"[" {System.out.println("Token [ <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

"]" {System.out.println("Token ] <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

"{" {System.out.println("Token { <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

"}" {System.out.println("Token } <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

[:digit:]+ {System.out.println("Token INT <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

[:digit:]\*'.'[0-9]+ {System.out.println("Token FLOAT <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

"$" {System.out.println("Token $ <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

[a-zA-Z\_][a-zA-Z0-9\_]\*[!|?] {System.out.println("Token IDFUNCTION <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

[a-zA-Z\_][a-zA-Z0-9\_]\* {System.out.println("Token Ident <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); identCount++;}

[@][a-zA-Z\_][a-zA-Z0-9\_]\* { System.out.println("Token Ident <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

{NL} { }

{TAB} { /\* ignora los tabuladores \*/ }

{BLANCO} { /\* ignora los espacios en blanco \*/ }

\" { yybegin(STRING); }

. {System.out.println("Token No Valido <" +yytext()+ ">linea: " + (yyline+1) + " columna: " + (yycolumn+1));}

}/\* fin YYinitial \*/

<IN\_COMMENT>{

“=end” { yybegin(YYINITIAL); }

[^=\n]

“=”

\n yylineo++;} {return COMMENT}

<STRING> {

"#{" { System.out.println("Token #{ <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1) );

}

([a-zA-Z0-9\.á-ú" "])+ { System.out.println("Token String <" +yytext()+ "> encontrado en linea: " + (yyline+1) + " columna: " + (yycolumn+1)); }

"\"" { yybegin(YYINITIAL); }

. { }

}