```
package Lexer;

import Parser.sym;
import AST.Decl;

%%
%{
private java_cup.runtime.Symbol tok(int k, Object value) {
//  System.out.println("Token: " + k);
  return new java_cup.runtime.Symbol(k, yyline, 0, value);
}
%}

%public
%cup
%line
%eofval{
{return tok(sym.EOF, null); }
%eofval}

letra= [a-zA-Z]


%%

";"             {return tok(sym.PUNTOCOMA, null); }
":="            {return tok(sym.ASOP, null); }
"+"             {return tok(sym.MAS, null); }
"="             {return tok(sym.IGUALQUE, null); }
"{"             {return tok(sym.ABRELLAVE, null); }
"}"             {return tok(sym.CIERRALLAVE, null); }
","             {return tok(sym.COMA, null); }
"("             {return tok(sym.PAREN, null); }
")"             {return tok(sym.TESIS, null); }
if              {return tok(sym.IF, null); }
then            {return tok(sym.THEN, null); }
endif           {return tok(sym.ENDIF, null); }
prog            {return tok(sym.PROG, null); }
in              {return tok(sym.IN, null); }
out             {return tok(sym.OUT, null); }
local           {return tok(sym.LOCAL, null); }
true            {return tok(sym.CLOG, new Boolean(true)); }
false           {return tok(sym.CLOG, new Boolean(false)); }
int             {return tok(sym.TIPO, new Boolean(Decl.tint)); }
bool            {return tok(sym.TIPO, new Boolean(Decl.tbool)); }
[0-9]+          {return tok(sym.CENT, new Integer(yytext())); }
{letra}({letra}|[0-9])* {return tok(sym.IDENT, yytext()); }
(" "|\n|\t|\r)+ { }
.               { System.out.println("Caracter Ilegal en linea" + yyline);}
```