Especificación Léxica

```
DIGITO = [0-9]
CARACTER =[A-Za-záéíóúÁÉÍÓÚñÑ_]
ComentariosLinea = "//" ~ \n
ComentariosMultiLinea = "/*" ~ "*/"
Real = ([0-9]* \.[0-9]+) | [0-9][eE][+-]?[0-9]+ | [0-9]+ \. [0-9]* [eE]-?[0-
9]+

%%
{ComentariosLinea}                          {}
{ComentariosMultiLinea}                     {}
dim                                         {return Parser.DIM;}
integer             {parser.setYylval(yytext()); return Parser.INTEGER;}
real                {parser.setYylval(yytext()); return Parser.REAL;}
character           {parser.setYylval(yytext()); return Parser.CHARACTER;}
as                                          {return Parser.AS;}
end                                         {return Parser.END;}
proc                                        {return Parser.PROC;}
type                                        {return Parser.TYPE;}
ctype                                       {return Parser.CTYPE;}
print                                       {return Parser.PRINT;}
while                                       {return Parser.WHILE;}
do                                          {return Parser.DO;}
if                                          {return Parser.IF;}
else                                        {return Parser.ELSE;}
then                                        {return Parser.THEN;}
function                                    {return Parser.FUNCTION;}
read                                        {return Parser.READ;}
return                                      {return Parser.RETURN;}
{Real}     {parser.setYylval(new Double(yytext())); return Parser.REAL;}
","                 {parser.setYylval(yytext()); return (int)yycharat(0);}
";"                 {parser.setYylval(yytext()); return (int)yycharat(0);}
[0-9]+   {parser.setYylval(new Integer(yytext())); return Parser.CTE_ENTERA;}
\'[^\']\'    {parser.setYylval(yytext().charAt(1)); return Parser.CHARACTER;}

"<="                {parser.setYylval(yytext()); return Parser.MENORIGUAL;}
">="                {parser.setYylval(yytext()); return Parser.MAYORIGUAL;}
"=="                {parser.setYylval(yytext()); return Parser.IGUALDAD;}
"<>"                {parser.setYylval(yytext()); return Parser.DISTINTO;}
"and"                                               {return Parser.AND;}
"or"                                                {return Parser.OR;}
"not"                                               {return Parser.NOT;}


")" |
"(" |
">" |
"<" |
"*" |
"/" |
"%" |
"-" |
"+" |
"=" |
"}" |
"{" |
```

```
"!" |
"[" |
"]" |
"." {parser.setYylval(yytext()); return (int)yycharat(0);}

{CARACTER}({CARACTER}|{DIGITO})*      {parser.setYylval(yytext()); return
Parser.IDENT;}

[ \n\r\t]                                          {}


.            { gestor.error("Error léxico en la línea " + line() + " y
columna " + column() + ".");}
```