```
%{
    /*
     * ========================================================================
     *
     * checkpoint2-tminet.l -- CS 4223 Checkpoint #2
     *
     * Programmer ------------ Thais Minet
     *
     * ========================================================================
     */

    #include <stdio.h>
    #include <string.h>
%}

%%
[mM][aA][iI][nN]                                          { printf("MAIN: %s\n", yytext); }

[dD][aA][tT][aA]                                          { printf("DATA: %s\n", yytext); }

[aA][lL][gG][oO][rR][iI][tT][hH][mM]                      { printf("ALGORITHM: %s\n", yytext); }

[iI][fF]                                                  { printf("IF: %s\n", yytext); }

[eE][lL][sS][eE]                                          { printf("ELSE: %s\n", yytext); }

[wW][hH][iI][lL][eE]                                      { printf("WHILE: %s\n", yytext); }

[cC][oO][uU][nN][tT][iI][nN][gG]                          { printf("COUNTING: %s\n", yytext); }

[uU][pP][wW][aA][rR][dD]                                  { printf("UPWARD: %s\n", yytext); }

[dD][oO][wW][nN][wW][aA][rR][dD]                          { printf("DOWNWARD: %s\n", yytext); }

[tT][oO]                                                  { printf("TO: %s\n", yytext); }

[eE][xX][iI][tT]                                          { printf("EXIT: %s\n", yytext); }

[eE][nN][dD]                                              { printf("END: %s\n", yytext); }

[rR][eE][aA][lL]                                          { printf("REAL: %s\n", yytext); }

[iI][nN][tT][eE][gG][eE][rR]                              { printf("INTEGER: %s\n", yytext); }

[pP][rR][iI][nN][tT]                                      { printf("PRINT: %s\n", yytext); }

[rR][eE][aA][dD]                                          { printf("REAL: %s\n", yytext); }

[a-zA-Z][a-zA-Z0-9]*                                      { printf("VARIABLE: %s\n", yytext); }

[-+]?(([0-9]+\.[0-9]+)|([0-9]+(\.[0-9]+)?[eE][-+][0-9]+)) { printf("REAL_CONST: %s\n", yytext); }

[0-9]+                                                    { printf("INTEGER_CONST: %s\n", yytext); }

\"([^"\n]|(\"\"))*["\n]                                   { printf("CHAR_STRING_CONST: %s\n", yytext); }

":"                                                       { printf("COLON: %s\n", yytext); }
```

```
":="                                                      { printf("ASSIGNMENT: %s\n", yytext); }

"&"                                                       { printf("AND: %s\n", yytext); }

"|"                                                       { printf("OR: %s\n", yytext); }

"~"                                                       { printf("NOT: %s\n", yytext); }

"<"                                                       { printf("LESS: %s\n", yytext); }

"<="                                                      { printf("LESS_EQUAL: %s\n", yytext); }

">"                                                       { printf("GREATER: %s\n", yytext); }

">="                                                      { printf("GREATER_EQUAL: %s\n", yytext); }

"="                                                       { printf("EQUAL: %s\n", yytext); }

"<>"                                                      { printf("NOT_EQUAL: %s\n", yytext); }

"+"                                                       { printf("ADD: %s\n", yytext); }

"-"                                                       { printf("SUB: %s\n", yytext); }

"*"                                                       { printf("MUL: %s\n", yytext); }

"/"                                                       { printf("DIV: %s\n", yytext); }

"%"                                                       { printf("MOD: %s\n", yytext); }

"("                                                       { printf("OPEN_PAREN: %s\n", yytext); }

")"                                                       { printf("CLOSE_PAREN: %s\n", yytext); }

"["                                                       { printf("OPEN_BRACKET: %s\n", yytext); }

"]"                                                       { printf("CLOSE_BRACKET: %s\n", yytext); }

#[^\n]*                                                   { printf("COMMENT: %s\n", yytext); }

"!"                                                       { printf("CARRIAGE_RETURN: %s\n", yytext); }

","                                                       { printf("COMMA: %s\n", yytext); }

\n                                                        { printf("NEWLINE: %s", yytext); }

";"                                                       { printf("END_STATEMENT: %s\n", yytext); }

[\t ]+                                                    { printf("WHITESPACE: %s\n", yytext); }
%%

int main() {
    yylex();
    return 0;
}
```