%option noyywrap

%x C\_COMMENT

%{

extern FILE \*yyin;

extern char \*yytext;

struct symbol {

char \*name;

int index;

};

struct text{

char \*name;

int index;

};

struct symbol sym[500];

struct text txt[500];

char \*temp;

int indexcount;

int id\_index();

void sym\_st();

void txt\_st();

int strcount;

int str\_index();

int yylval;

char\* trim();

%}

letter [A-Za-z\_]

ws [ \t\n]+

digit [0-9]

id {letter}({letter}|{digit})\*

INTEGER [-+]?{digit}+

DOUBLE [+-]?((([0-9]+)|([0-9]\*\.[0-9]+))([eE][-+]?[0-9]+)?)

string \"(\n\\.|[^"])\*\"

linecomment \/\/(.\*)

comment \/\\*( [^\*] | \\*+ [^\*/] )\*\\*+\/

%%

{ws} {}

[\_]+ {printf("!!!!!!l\_\_lexical error underbar\_ %s\n",yytext);}

[$|#|@|%|&]+ {printf("!!!!!!!\_\_lexical error specaisl %s \n", yytext);}

int {printf("<%10s, > %10s\n", "INT", yytext);}

double {printf("<%10s, > %10s\n", "DOUBLE", yytext);}

str {printf("<%10s, > %10s\n", "STR", yytext);}

if {printf("<%10s, > %10s\n", "if", yytext);}

while {printf("<%10s, > %10s\n", "while", yytext);}

return {printf("<%10s, > %10s\n", "return", yytext);}

"+" { printf("<%10s, > %10s\n","PLUS", yytext); }

"-" { printf("<%10s, > %10s\n","MINUS", yytext); }

"\*" { printf("<%10s, > %10s\n","MULT", yytext); }

"/" { printf("<%10s, > %10s\n","DIV", yytext); }

"=" { printf("<%10s, > %10s\n","ASSIGN", yytext); }

">" { printf("<%10s, > %10s\n","GRATER", yytext); }

"<" { printf("<%10s, > %10s\n","LESSER", yytext); }

">=" { printf("<%10s, > %10s\n","GEQ", yytext); }

"<=" { printf("<%10s, > %10s\n","LEQ", yytext); }

"==" { printf("<%10s, > %10s\n","EQEL", yytext); }

"!=" { printf("<%10s, > %10s\n","NEQ", yytext); }

"\"" { printf("<%10s, > %10s\n","MLCS", yytext); }

"," { printf("<%10s, > %10s\n","COMMA", yytext); }

"(" { printf("<%10s, > %10s\n","LPAREN", yytext); }

")" { printf("<%10s, > %10s\n","RPAREN", yytext); }

";" { printf("<%10s, > %10s\n","SEMICOL", yytext); }

"{" { printf("<%10s, > %10s\n","LBRACE", yytext); }

"}" { printf("<%10s, > %10s\n","RBRACE", yytext); }

{id} {

yylval = id\_index();

if(yylval == indexcount) sym\_st();

printf("<%10s, %d> %10s\n","ID", yylval, sym[yylval].name);

}

{INTEGER} { if(strlen(yytext)>10){

temp = trim();

printf("<%10s, %s>\n", "INTEGER",temp );}

else printf("<%10s, %s>\n", "INTEGER", yytext); }

{DOUBLE} { printf("<%10s, %s>\n", "DOUBLE", yytext); }

{string} { yylval = str\_index();

if(yylval == strcount) txt\_st();

printf("<%10s, %d> \t%s\n", "STRING",yylval, yytext);}

{linecomment} {printf("<%10s> %s\n","COMMENT", yytext);}

. {}

%%

int main(int argc, int \*argv[]){

indexcount=1;

strcount=1;

printf("%s %20s","Token", "Lexeme\n");

if(argc<2){

// printf("input is file\n");

yylex();

}

else{

yyin = fopen(argv[1], "r");

yylex();

}

printf("=============SYMBOL TABLE==========\n");

for(int i=1; i< indexcount; i++){

printf("%d %10s\n", sym[i].index, sym[i].name);

}

printf("==============STRING TABLE=============\n");

for(int i=1; i< strcount; i++){

printf("%d %10s\n", txt[i].index, txt[i].name);

}

}

int id\_index(){

for(int i=1; i < indexcount; i++){

//printf("%d %d\n", sizeof(yytext), sizeof(sym[i].name));

if(strncmp(yytext,sym[i].name,16)==0){

// printf("동일\n");

return i;

}

}

return indexcount;

}

void sym\_st(){

// printf("size ==%d", strlen(yytext));

if(strlen(yytext) >16){

sym[yylval].name = strndup(yytext,16);

}

else{

sym[yylval].name = strdup(yytext);

}

sym[yylval].index = yylval;

indexcount++;

}

int str\_index(){

for(int i=1; i < strcount; i++){

if(strcmp(yytext,txt[i].name)==0){

return i;

}

}

return strcount;

}

void txt\_st(){

txt[yylval].name = strdup(yytext);

txt[yylval].index = yylval;

strcount++;

}

char\* trim(){

int a = strlen(yytext)-10;

temp = strdup(yytext+a);

return temp;

}