A2 Lex Specification: scan.l

```
digit           [0-9]
letter          [a-zA-Z_]
delim           [ ₩t]
line            [₩n]
ws              {delim}+

%{
#include "y.tab.h"
#include "type.h"
extern int yylval;
extern int line_no;
extern A_ID *current_id;
char    *makeString();
int     checkIdentifier();
%}
%%
{ws}            { }
{line}          { line_no++;}
auto            { return(AUTO_SYM); }
break           { return(BREAK_SYM); }
case            { return(CASE_SYM); }
continue        { return(CONTINUE_SYM); }
default         { return(DEFAULT_SYM); }
do              { return(DO_SYM); }
else            { return(ELSE_SYM); }
enum            { return(ENUM_SYM); }
for             { return(FOR_SYM); }
if              { return(IF_SYM); }
return          { return(RETURN_SYM); }
sizeof          { return(SIZEOF_SYM); }
static          { return(STATIC_SYM); }
struct          { return(STRUCT_SYM); }
```

```
switch          { return(SWITCH_SYM); }
typedef         { return(TYPEDEF_SYM); }
union           { return(UNION_SYM); }
while           { return(WHILE_SYM); }
"₩+₩+"          { return(PLUSPLUS); }
"₩-₩-"          { return(MINUSMINUS); }
"₩->"           { return(ARROW); }
"<"             { return(LSS); }
">"             { return(GTR); }
"<="            { return(LEQ); }
">="            { return(GEQ); }
"=="            { return(EQL); }
"!="            { return(NEQ); }
"&&"            { return(AMPAMP); }
"||"            { return(BARBAR); }
"₩.₩.₩."        { return(DOTDOTDOT); }
"₩("            { return(LP); }
"₩)"            { return(RP); }
"₩["            { return(LB); }
"₩]"            { return(RB); }
"₩{"            { return(LR); }
"₩}"            { return(RR); }
"₩:"            { return(COLON); }
"₩."            { return(PERIOD); }
"₩,"            { return(COMMA); }
"₩!"            { return(EXCL); }
"₩*"            { return(STAR); }
"₩/"            { return(SLASH); }
"₩%"            { return(PERCENT); }
"₩&"            { return(AMP); }
"₩;"            { return(SEMICOLON); }
"₩+"            { return(PLUS); }
"₩-"            { return(MINUS); }
"₩="            { return(ASSIGN); }
```

```
{digit}+                    { yylval=atoi(yytext); return(INTEGER_CONSTANT);}
{digit}+₩.{digit}+          { yylval=makeString(yytext); return(FLOAT_CONSTANT);}
{letter}({letter}|{digit})* { return(checkIdentifier(yytext)); }
₩"([^"₩n]|₩₩["₩n])*₩"       { yylval=makeString(yytext); return(STRING_LITERAL);}
₩'([^'₩n]|₩'₩')₩'           { yylval=*(yytext+1); return(CHARACTER_CONSTANT);}
"//"[^₩n]*                  { }
%%
char *makeString(char *s) {
        char *t;
        t=malloc(strlen(s)+1);
        strcpy(t,s);
        return(t);
}
int checkIdentifier(char *s) {
        A_ID *id;
        char *t;
        id=current_id;
        while (id) {
                if (strcmp(id->name,s)==0)
                        break;
                id=id->prev; }
        if (id==0) {
                yylval=makeString(s);
                return(IDENTIFIER); }
        else if (id->kind==ID_TYPE) {
                yylval=id->type;
                return(TYPE_IDENTIFIER); }
        else {
                yylval=id->name;
                return(IDENTIFIER);}
}
```