```
%{
/*
Vahe Karamian - CS 440 - Project 1
Filename: project1.l, project1.h, project1.c
*/

#include <stdio.h>
#include <math.h>

/* this function I use for my symbol table */
void SymbolTable( char* id );

/* global index variable used by the symbol table */
int  vk_index = 0;

/* symbol table data structure */
char **symbolTable[100];

/* declare enum for the keywords and etc... */
enum{IF=10001,ELSE,WHILE,BREAK,RETURN,VOID,INT};
enum{ADDOP=20001,SUBOP,MULOP,DIVOP};
enum{LTOP=30001,GTOP,EQOP,LEOP,GEOP,NEOP,ASOP};
enum{SEMI=40001,COMA};
enum{LPAREN=50001,RPAREN,LBRACE,RBRACE,LBRAKT,RBRAKT};

%}

/* digit, letter, comment */
DIGIT [0-9]
LETTER  [A-Za-z]
%x    COMMENT

%%

/* begining of comment section detection */
"/*"  { BEGIN COMMENT; }  /* enter comment eating state */
"/*".*"*/"[ \t]*\n  { ; } /* self contained comment */

<COMMENT>"*/"[ \t]*\n { BEGIN 0; }
<COMMENT>"*/"     { BEGIN 0; }
<COMMENT>\n       { ; }
<COMMENT>.\n      { ; }
/* end of comment section detection */

/* begining of keyword detection */
if    printf("(t_if [%d])", IF );
else  printf("(t_else [%d])", ELSE );
while printf("(t_while [%d])", WHILE );
break printf("(t_break [%d])", BREAK );
return  printf("(t_return [%d])", RETURN );
void  printf("(t_void [%d])", VOID );
int   printf("(t_int [%d])", INT );
/* end of keyword detection */

/* begining of operator detection */
"+" printf("(t_ADD %d)", ADDOP );
"-" printf("(t_SUB %d)", SUBOP );
"/" printf("(t_DIV %d)", DIVOP );
"*" printf("(t_MUL %d)", MULOP );
/* end of operator detection */

/* begining of relational operator detection */
"<" printf("(t_LT %d)", LTOP );
">" printf("(t_GT %d)", GTOP );

"=="  printf("(t_EQ %d)", EQOP );
"<="  printf("(t_LE %d)", LEOP );
">="  printf("(t_GE %d)", GEOP );
"!="  printf("(t_NE %d)", NEOP );

"=" printf("(t_AS %d)", ASOP );
/* enf od relational operation detection */

";" printf("(t_semi %d)\n", SEMI );
"," printf("(t_coma %d)", COMA );

"(" printf("(t_lparen %d)", LPAREN );
")" printf("(t_rparen %d)", RPAREN );
"{" printf("(t_lbrace %d)\n", LBRACE );
"}" printf("(t_rbrace %d)\n", RBRACE );
"[" printf("(t_lbrakt %d)", LBRAKT );
"]" printf("(t_rbrakt %d)", RBRAKT );

/* detecting digits */
{DIGIT}+            printf("(t_num %s)", yytext );

/* detecting floats */
{DIGIT}+"."{DIGIT}*       printf("(t_float %s)", yytext );

/* detecting the identifiers */
{LETTER}({LETTER}|{DIGIT})* SymbolTable( yytext );

[ \t\n]  ; /* eat up white spaces */

/* unrecognized character */
.             printf("(\nt_error: %s\n)", yytext );

%%

/* symbol table function */
void SymbolTable( char* id )
{
  symbolTable[vk_index] = id;
  printf("(t_id {%s} L[%i])", symbolTable[vk_index], vk_index );
  vk_index++;
}

/* main function */
int main( int argc, char *argv[] )
{
  int vv=0;
  ++argv, --argc;
  if(argc>0)
    yyin = fopen(argv[0], "r");
  else
    yyin = stdin;

  yylex( );
}
```