.l File

```
%{
#include "y.tab.h"
void yyerror (char *s);
int yylex();
%}
%%
[ \t\n] ;
"print" {return PRINT;}
"exit" {return STOP;}
[a-zA-Z] {yylval.id = yytext[0]; return ID;}
[0-9]+ {yylval.num = atoi(yytext); return NUM;}
[-+*/%=;] {return yytext[0];}
. {yyerror ("unexpected character");}

%%
int yywrap (void)
{
return 1;
}
```

.y File

```
%{
void yyerror (char *s);
int yylex();
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
extern FILE *yyin;
extern int yylineno;
int sym_table[52];
int symNumVal(char symbol);
void update(char symbol, int val);
extern int yylex();
extern int yywrap();
extern int yyparse();

%}

%union {int num; char id;}
%token PRINT STOP
```

```
%token <id> ID
%token <num> NUM
%type <num> S EXP TERM
%type <id> ASSIGN
%%
S : ASSIGN ';' {;}
| STOP ';' {exit(EXIT_SUCCESS);}
| PRINT EXP ';' {printf("%d\n", $2);}
| S S
;
ASSIGN : ID '=' EXP { update($1,$3); }
;
EXP : TERM {$$ = $1;}
| EXP '+' EXP {$$ = $1 + $3;}
| EXP '-' EXP {$$ = $1 - $3;}
| EXP '*' EXP {$$ = $1 * $3;}
| EXP '/' EXP {$$ = $1 / $3;}
| EXP '%' EXP {$$ = $1 % $3;}
;
TERM : NUM {$$ = $1;}
| ID {$$ = symNumVal($1);}
;
%%
int getSymIndex(char token)
{
int idx = -1;
if(islower(token)) {
idx = token - 'a' + 26;
} else if(isupper(token)) {
idx = token - 'A';
}
return idx;
}
int symNumVal(char symbol)
{
int a = getSymIndex(symbol);
return sym_table[a];
}
void update(char symbol, int val)
{
int a = getSymIndex(symbol);
sym_table[a] = val;
}
int main (void)
```

```
{
for(int i=0; i<52; i++)
{
sym_table[i] = 0;
}
return yyparse ( );
}
void yyerror (char *s)
{
fprintf (stderr, "%s\n", s);
}
```

Output :