Lex

```
%{
#include "calc.tab.h" // Include the BISON header
%}
/* Token Definitions */
%%
[0-9]+  { yylval = atoi(yytext); return NUMBER; }
"+"     { return ADD; }
"-"     { return SUB; }
"*"     { return MUL; }
"/"     { return DIV; }
"("     { return LPAREN; }
")"     { return RPAREN; }
[\t ]+   ; // Ignore whitespace
\n      { return '\n'; }
.       { printf("Unexpected character: %s\n", yytext); }
%%
// Function to handle end-of-file
int yywrap() {
   return 1;
}
\\COMMENTS
bison -d calc.y
flex calc.l
gcc lex.yy.c calc.tab.c -o calculator
./calculator
5 + 3 * 2
```

```
%{
#include <stdio.h>
%}

%%
"int"|"float"|"char"|"double"|"return"|"if"|"else"|"while"|"for"|"void" {
   printf("Keyword: %s\n", yytext);
}
[a-zA-Z_][a-zA-Z0-9_]* {
   printf("Identifier: %s\n", yytext);
}
[0-9]+ {
   printf("Number: %s\n", yytext);
}
"=="|"!="|"<="|">="|"&&"|"||"|"!" {
   printf("Logical Operator: %s\n", yytext);
}
"+"|"-"|"*"|"/"|"=" {
   printf("Arithmetic Operator: %s\n", yytext);
}
"("|")"|"{"|"}"|";" {
   printf("Symbol: %s\n", yytext);
}
[ \t\n] ;  // Ignore whitespaces and newlines
. {
   printf("Unknown Character: %s\n", yytext);
}
%%

int main() {
   printf("Enter code: \n");
   yylex();
   return 0;
}

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

LEXICAL

BISON

```
%{
#include <stdio.h>
#include <stdlib.h>
extern int yylex();
void yyerror(const char *s);
%}
%token NUMBER ADD SUB MUL DIV LPAREN RPAREN
%%
calculation:
    calculation expression '\n' { printf("Result: %d\n", $2); }
    | /* Empty */ ;
expression:
    expression ADD term { $$ = $1 + $3; }
    | expression SUB term { $$ = $1 - $3; }
    | term { $$ = $1; }
    ;
term:
    term MUL factor { $$ = $1 * $3; }
    | term DIV factor {
        if ($3 == 0) {
            yyerror("Division by zero");
            exit(1);
        }
        $$ = $1 / $3;
    }
    | factor { $$ = $1; }
    ;
factor:
    NUMBER { $$ = $1; }
    | LPAREN expression RPAREN { $$ = $2; }
    ;
%%
void yyerror(const char *s) {
    fprintf(stderr, "Error: %s\n", s);
}
int main() {
    printf("Enter an arithmetic expression:\n");
    return yyparse();
}
```