```
// $Id: extern.h,v 1.1 2018-04-02 15:31:00-07 - - $

#ifndef __EXTERN_H__
#define __EXTERN_H__

int yylex();
int yyparse();

void badchar (char);
void yyerror (const char*);

extern char* yytext;

#define YYSTYPE double
#include "yyparse.h"

#endif

```

```
/* $Id: lexer.l,v 1.1 2018-04-02 15:31:00-07 - - $ */

%{

#include "extern.h"

%}

%option noyywrap

MANTISSA    ([[:digit:]]+\.?[[:digit:]]*|\.[[:digit:]]+)
EXPONENT    ([Ee][+-]?[[:digit:]]+)
INFINITY    ([Ii][Nn][Ff][[:isalnum:]]*)
NAN         ([Nn][Aa][Nn][[:isalnum:]]*)
NUMBER      ({MANTISSA})({EXPONENT})?|INFINITY|NAN


%%

"#".*
[ \t]+
\n         { return '\n'; }
{NUMBER}   { return NUMBER; }
"+"        { return '+'; }
"-"        { return '-'; }
"*"        { return '*'; }
"/"        { return '/'; }

%%
```

```
// $Id: parser.y,v 1.1 2018-04-02 15:31:00-07 - - $

%{

#include <stdio.h>
#include <stdlib.h>

#include "extern.h"

%}

%token NUMBER

%%

input : input expr '\n' { printf ("%.15g\n", $2); }
      | '\n'            { printf ("\n"); }
      | error '\n'      { printf ("error\n"); }
      |
      ;

expr  : NUMBER          { $$ = atof (yytext); }
      | expr expr '+'   { $$ = $1 + $2; }
      | expr expr '-'   { $$ = $1 - $2; }
      | expr expr '*'   { $$ = $1 * $2; }
      | expr expr '/'   { $$ = $1 / $2; }
      ;

%%

```

```
// $Id: main.cpp,v 1.1 2018-04-02 15:31:00-07 - - $

#include <stdio.h>
#include <stdlib.h>

#include "extern.h"

int status = EXIT_SUCCESS;

void yyerror (const char* error) {
   printf ("%s\n", error);
   status = EXIT_FAILURE;
}

int main () {
   yyparse();
   return status;
}
```

```
# $Id: Makefile,v 1.1 2018-04-02 15:31:00-07 - - $

CPP    = g++ -g -O0 -std=gnu++17 -Wall -Wextra -Wold-style-cast
CPPYY  = g++ -g -O0 -std=gnu++17 -Wno-register
OBJS   = main.o yyparse.o yylex.o
GENS   = yyparse.h yyparse.cpp yylex.cpp
SRCS   = extern.h lexer.l parser.y main.cpp Makefile

all : rpn

rpn : ${OBJS}
	${CPP} -o rpn ${OBJS}

%.o : %.cpp
	checksource $<
	cpplint.py.perl $<
	${CPP} -c $<

yyparse.cpp yyparse.h : parser.y
	bison --defines=yyparse.h --output=yyparse.cpp parser.y

yyparse.o : yyparse.cpp
	${CPPYY} -c yyparse.cpp

yylex.cpp : yyparse.h lexer.l
	flex --outfile=yylex.cpp lexer.l

yylex.o : yylex.cpp
	${CPPYY} -Wno-register -c yylex.cpp

spotless : clean
	- rm rpn Makefile.deps Listing.{ps,pdf}

clean :
	- rm ${OBJS} ${GENS}

ci :
	- cid + ${SRCS}

lis : ${SRCS}
	mkpspdf Listing.ps ${SRCS}

Makefile.deps : yyparse.cpp yylex.cpp
	${CPP} -MM yyparse.cpp yylex.cpp main.cpp

include Makefile.deps

```