Practical4.y

%{

#include <stdio.h>

#include <math.h>

int yylex(void);

void yyerror(char \*message);

%}

%union {

double num;

}

%token <num> NUMBER

%left '+' '-'

%left '\*' '/'

%right '^'

%type <num> expr

%%

input

: /\* empty \*/

| input line

;

line

: '\n'

| expr '\n' { printf("Answer = %g\n", $1); }

;

expr

: expr '+' expr { $$ = $1 + $3; }

| expr '-' expr { $$ = $1 - $3; }

| expr '\*' expr { $$ = $1 \* $3; }

| expr '/' expr {

if ($3 == 0.0) {

yyerror("division by zero");

$$ = 0.0;

} else {

$$ = $1 / $3;

}

}

| expr '^' expr { $$ = pow($1, $3); }

| '(' expr ')' { $$ = $2; }

| '-' expr { $$ = -$2; }

| NUMBER { $$ = $1; }

;

%%

void yyerror(char \*message)

{

fprintf(stderr, "Error: %s\n", message);

}

int main(void)

{

return yyparse();

}

Practical4.l

%{

#include "y.tab.h"

int ans;

%}

%%

[0-9]+(\.[0-9]+)?    { yylval.num = atof(yytext); return NUMBER; }

[+\-\*/^()]           { return yytext[0]; }

\n                   { return '\n'; }

[ \t]                ;

.                    { printf("Invalid character: %s\n", yytext); }

%%

int yywrap(void)

{

    return 1;

}

