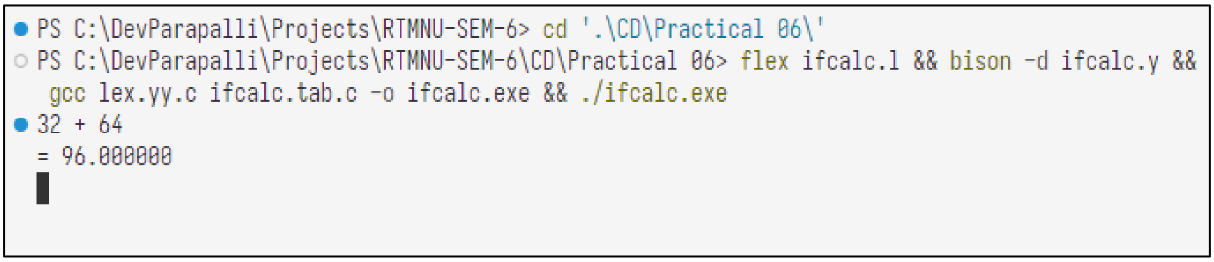
ifcalc.y

```
%{
    #define YYSTYPE double

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

    int yylex(void);
    int yyerror(char *);
%}

%token NUM LPAREN RPAREN

%left '+' '-'
%left '*' '/'
%right '^'

%%

input:   /* empty */
    | input line
    ;

line: '\n'
    | exp '\n' { printf("= %f\n", $1); }
;

exp: NUM { $$ = $1; }
    | LPAREN exp RPAREN { $$ = $2; }
    | exp '+' exp { $$ = $1 + $3; }
    | exp '-' exp { $$ = $1 - $3; }
    | exp '*' exp { $$ = $1 * $3; }
    | exp '/' exp { $$ = $1 / $3; }
    | exp '^' exp { $$ = pow($1, $3); }
;

%%

int yyerror(char *s) {
    fprintf(stderr, "![E] %s\n", s);
    return 0;
}

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

ifcalc.l

```
%option noyywrap
%{
    #include <stdlib.h>
    #include <stdio.h>
    #include <string.h>
    #include <math.h>

    #define YYSTYPE double

    #include "ifcalc.tab.h"

    void yyerror(char *);

%}

%%

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

"(" {return LPAREN;}
")" {return RPAREN;}

[+*^/\-] | /* Send operator as is */

"\n" {
    return *yytext;
}

[ \t]+ {
    /* ignore whitespace */
}

%%
```

Output



```
● PS C:\DevParapalli\Projects\RTMNU-SEM-6> cd '.\CD\Practical 06\'
○ PS C:\DevParapalli\Projects\RTMNU-SEM-6\CD\Practical 06> flex ifcalc.l && bison -d ifcalc.y &&
   gcc lex.yy.c ifcalc.tab.c -o ifcalc.exe && ./ifcalc.exe
● 32 + 64
  = 96.000000
```