Parser Implementation
(Due 5/20)

Please write a recursive descent parser (including a lexical analyzer) for the following EBNF in your favorite programming language (such as C, C++, Java, etc.). Your program codes should be runnable.

<exprs> -> <expr>; {<expr> ;}
<expr> -> <term> { (+|-) <term> }
<term> -> <factor> { (*|/) <factor> }
<factor> -> <exp> { ^ <exp> }
<exp> -> id | int_lit | real_lit | (<expr>)

where,
^ indicates the power operation, id is a legal identifier name, int_lit represents any positive integer number, and real_lit represents any positive floating-point number.

Input and output examples:

Input:
x ^ (y+1) - x/2.5 + z;
sum + total * 10;
Output:
parse succeed
Input:
3 + x*y);
Output:
parse fail

Note 1: Please name your input file as “input.txt”
Note 2: Add some error messages in output such as “lack of right parenthesis” for getting a high score.
Note 3: Try to rewrite the grammar to allow negative numbers for getting a high score.