CIIC 4030/ICOM 4036 Programming Languages
Assignment #2

Given the lexical analyzer implemented in assignment #1, add the parsing code using PLY to recognize the following context-free grammar:

```
global_facts -> facts exec_line

facts -> func_def facts
        | assign   facts
        | empty

func_def -> FUNC ID_FUNC LBRACE params RBRACE ASSIGN stm END

params -> ID_FUNC COMMA params
        | ID COMMA params
        | ID_FUNC
        | ID

assign -> VAL ID ASSIGN stm END

stm -> ID_FUNC LBRACE args RBRACE

args -> ID_FUNC COMMA args
        | stm COMMA args
        | ID_FUNC
        | stm

stm -> stm PLUS stm
        |stm MINUS stm
        | stm TIMES stm
        | stm DIVIDE stm
        | stm DOT stm
        | stm LESSTHAN stm
        | stm GREATERTHAN stm
        | stm EQUAL stm
        | stm AND stm
        | stm OR stm
        |STRING
        |NUMBER
        |TRUE
        |FALSE
        |NIL
        |ID
        |LPAREN stm RPAREN
        |IF stm THEN stm ELSE stm END
        |LET facts IN stm END

exec_line -> EXEC stm
```

Instructions:

1. Write function definitions for each BNF grammar rule.
2. Define operators' precedence to reduce language ambiguity. Be sure the precedence maintains the operator hierarchy you already know from traditional arithmetic and logical operations.
3. Note that there is a difference between identifiers (ID) and function names (ID_FUN). While ID begins with lower case, ID_FUNC starts with upper case. Modify the lexical definition to tokenize those identifiers appropriately.
4. Note that some rules have multiple options. Separating rules is highly recommended.
5. Define an error function that prints a message: "Syntax error in input: X" indicating the line of code where the error appears. With no syntax error in the program, it should print "Syntax error in input: none."