Objectives:
1- To develop a recursive-descent parser for a given grammar.
2- To generate a syntax tree as an output of the parser
3- To handle syntax errors.

C- Language
The language that will be used in this assignment is the language called C- defined in Appendix A of your text. One simple change to the grammar is to add the keyword “let” in production rule number 18 as follows:

18. expression \( \rightarrow \) let var = expression | simple-expression

Covert the grammar to be in EBNF as it is easier to write the recursive descent parser when the grammar is written in EBNF.

It is also possible to keep in BNF but you have to remove left recursion and left factor the rules that have common prefix.

Syntax Error Handling:
If a syntax error occurs while parsing, an error message should be reported specifying the line number, the look-ahead token that caused the error, and the expected token(s). Compilation can terminate at the first syntax error. Error recovery is not required.

Output the Parser:
The input source program augmented with syntactic errors either after the line in which the error occurs or at the end of the program. In case that there are no errors the parser output must be the syntax tree represented in a listed form. For example, the following syntax tree:

```
+   
/    
A     
|     
B     
|     
C
```
can be represented as (+ (A (* ( B C))). So what you can do is to generate the syntax tree as explained in the lecture, generate the tree represented as a list by preorder traversal of the tree, and then output the list.

Report:
A detailed report should be written explaining your design and implementation. Specifically, you need to discuss:
- The grammar after removing left recursion and common prefixes
- Implementation Issues and decisions
- Handling syntax errors, and
- Sample Runs of Test cases

To submit by email:
1. The report document.
2. All source files and executable file

**Grading**

Your grade will be divided into the following components:
1. Correctness and output 70%
2. Grammar completeness and correctness 10%
3. Handling syntax errors, and error reporting 10%
4. Report Document 10%