Principles of Programming Languages [PLP]
Exercises on Parsing

1) Consider the following grammar **G** :

$$\mathbf{R \rightarrow (R) \ | \ R + R \ | \ R R \ | \ R^* \ | \ a}$$

   a) Provide a leftmost and a rightmost derivation for string **(a+a)*a**
   b) Draw a parse tree for string **(a+a)*a**
   c) Describe the language generated by grammar **G**
   d) Draw an abstract syntax tree of string **(a+a)*a**
   e) Is grammar **G** ambiguous? Is string **(a+a)*a** ambiguous?
   f) Transform grammar **G** by left-factorizing it: call *LF*(**G**) the resulting grammar
   g) Draw a parse tree for the string **(a+a)*a** with respect to grammar *LF*(**G**)
   h) Transform grammar **G** by eliminating left recursion, obtaining grammar *LRE*(**G**)
   i) Draw a parse tree for the string **(a+a)*a** with respect to grammar *LRE*(**G**)

2) Consider the following grammar over the set of terminal symbols **{id , " , + }**:

$$\begin{aligned} &\mathbf{S} \rightarrow \mathbf{id} \ | \ \mathbf{" \ T \ "} \\ &\mathbf{T} \rightarrow \mathbf{S \ V} \\ &\mathbf{V} \rightarrow \varepsilon \ | \ \mathbf{+ \ S \ V} \end{aligned}$$

   a) Show First(α) for each production X $\rightarrow$ α and Follow(A) for each non-terminal A
   b) Build the LL(1) parse table
   c) Starting from the configuration (**stack:** $\boxed{\text{S \$}}$, **input:** $\boxed{\text{" id + id " \$}}$), show the evolution of the stack and of the input **in the first six steps** of the top-down predictive parsing algorithm using the LL(1) parse table. (Note: the top of the stack is to the left.)

3) Given grammar $\quad$ **A → A A + | a**
   a) Is string **Aa+A+** a sentential form? Is it a right-sentential form?
   b) Which is the handle in string **AA+a+** ?

4) Given grammar $\quad$ **E → E + E | x**
   a) Is the following claim true or false? Motivate your answer.
      "In string **E+E+x** , both **E + E** and **x** are handles."

5) Consider grammar

$$S \rightarrow (A) \mid x$$
$$A \rightarrow A + S \mid S$$

a) Show First(α) for each production $X \rightarrow \alpha$ and Follow(X) for each non-terminal X
b) Is the grammar LL(1)? Justify your answer
c) Draw the LR(0) automaton of the grammar
d) Draw the SLR parsing table of the grammar
e) Starting from the configuration (**stack:** $\boxed{0}$, **input:** $\boxed{(x + x)}$), show the evolution of the stack and of the input **in the first six steps** of the bottom-up LR parsing algorithm using the SLR(1) parse table. (Note: the "0" in the stack represents the start state of the LR(0) automaton.)

6) Consider the following grammar, whose terminals are {**a**, **?**}:

$$\mathbf{S \rightarrow A}$$
$$\mathbf{A \rightarrow B \mid B\ A}$$
$$\mathbf{B \rightarrow a\ ?\ C}$$
$$\mathbf{C \rightarrow \varepsilon \mid a\ C}$$

a) Left-factor the grammar,
b) Compute the First(A) and Follow(α) sets for each production A → α of the resulting grammar.
c) Build the LL(1) parse table.
d) Explain why the grammar is not LL(1).
e) Show that the language is LL(2), arguing convincingly that the conflicts can be resolved by looking ahead one more token.

7) Consider the grammar:

$$A \rightarrow CaBa$$
$$A \rightarrow B$$
$$B \rightarrow C$$
$$C \rightarrow b$$

a) What is the language generated by the grammar? Is it ambiguous?
b) Construct the LR(0) automaton
c) Build the SLR parse table. Is the grammar SLR?
d) Construct the LR(1) automaton and the LR(1) parsing table. Is the grammar LR(1)?