1 Left recursion

Consider the following grammar.

\[
\begin{align*}
\text{<sentence>} & ::= \text{<words>} \\
\text{<words>} & ::= \text{<words><word>} | \text{<word>}
\end{align*}
\]

Do the following tasks.

1. Remove the left recursion.

2. Why are left recursions bad? For what type of parsers? Keep the answer short and precise.

2 Extending the grammar

Extend the grammar from Exercise 1 so it can support questions (sentences terminated with a question mark), exclamations (sentences terminated with an exclamation mark), complex sentences (parts are divided by a comma), and the notion that the first word of a sentence must begin with a capital letter. Also, any other word in the sentence can begin with a capital letter. Assume that \text{<capitalWord>} is a word with a capital first letter.

Extra task (not graded): Write regular expressions for \text{<capitalWord>} and \text{<word>}.

3 Parsing mathematical expressions in reverse Polish notation

Write a grammar for parsing mathematical expressions in Reverse Polish Notation (RPN). Your grammar should be able to parse expressions that include digits the binary operators * (multiplication), - (subtraction), + (addition), / (division), as well as the unary operators N (negation) and ! (factorial).

Valid expressions:

- \(3 4 +\)
- \(10 2 + ! 3 4 + * 5 6 * /\)
- \(2 ! 3 ! +\)
- \(12 N\)
- \(12 13 -\)
- \(1 2 N - ! N\)

Invalid expressions:

\(^1\text{https://en.wikipedia.org/wiki/Reverse_Polish_notation}\)
• $3 + 4$
• $+ 1 2$
• $1 +$
• $5 -$

Extra task (not graded): Ensure that your grammar does not contain any left recursion.