Mini Assignment - Definite Clause Grammar

Instructions: The intent here is for you to try out some basics before you attempt the main assignment. The solution has been provided in class. Give it a good shot before you look into the provided solution. If you look into the solution, then understand what you were doing wrong and then try to write the code on your own. You must write the code in SWI **prolog** and submit it with at least two sample runs for each problem.

You should **submit one PDF file**. This PDF file should contain **code to the solution** of the problems (in text format only, no snapshot will be accepted) and **your program's sample runs** with output (in text format only, no snapshot will be accepted). Make sure to paste your code correctly (with correct indentation).

---

1. **(3 points)** Write a parser to parse a sentence by encoding the following Context Free Grammar (CFG) using Definite Clause Grammar (DCG). You can pass the input sentence as a list of tokens. Your parser should return 'true' for valid input sentences and 'false' otherwise.

(1) *sentence* → *noun-phrase verb-phrase* .
(2) *noun-phrase* → *article noun*
(3) *article* → `a` | `the`
(4) *noun* → `girl` | `dog`
(5) *verb-phrase* → *verb noun-phrase*
(6) *verb* → `sees` | `pets`

Figure 6.2 A grammar for simple english sentences

2. **(3 points)** Modify the parser created for Problem 1 to output a parse tree for valid input sentences.

3. **(4 points)** Eliminate left recursion in the following grammar and implement it using DCG.

expr → *expr* + *expr* | *expr* * *expr* | (*expr*) | *number*
number → *number digit* | *digit*
digit → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Figure 6.4 A simple integer arithmetic expression grammar
