Homework: Logic Programming

Instructions:

- Total points 48 pt
- Early deadline: Nov 28 (Wed) 2018 at 6:00 PM; Regular deadline: Nov 30 (Fri) 2018 at 6:00 PM (or till TAs start grading the homework)
- Download and install Swi-prolog `http://www.swi-prolog.org/`
- In this homework, you will write prolog programs for a set of given problems. The goal is to help you learn programming in Prolog and thinking in logic programming paradigm.
- Please zip .pl files and output files for all the solutions and submit it to Canvas.

Learning Objectives:

1. Problem solving using logic programming paradigm
2. Prolog programming

Questions:

1. (3 pt) Understand the following Prolog program:

   Given: $mystery([\,], L2, L2)$.
   $mystery([H|Tail], L2, [R|RTail]) :-$
   $H = R$,
   $mystery(Tail, L2, RTail)$.

   What would Z be in $mystery([3,3,2],[5,10,6],Z)$.

   Sol. Z = [3, 3, 2, 5, 10, 6].

2. (10 pt) Prolog programming:
   - (5 pt) Compute a factorial of a list of numbers.
   - (5 pt) `nextto(X, Y, List)` returns *true* if Y directly follows X in the list, else returns *false*. For example:

```
?- nextto(banana, apple, [apple, banana]).
false.
?- nextto(banana, apple, [grape, banana, apple]).
true.
```

Sol

- Factorial

```
factorial(0,1).

factorial(N,F) :-
  N > 0,
  N1 is N-1,
  factorial(N1,F1),
  F is N * F1.
```

- nextto

```
nextto(A, B, Ls) :- append(_, [A, B | _], Ls).
```

3. (15 pt) Write a Prolog program for parsing:

   (a) (8 pt) Consider the simple grammar below. Write a Prolog program that parses sentences (represented as lists of words) using the grammar. This grammar states that a sentence consists of a noun phrase, followed by a verb phrase, followed by a period. It also states that an article is either the word "a" or the word "the". Hint: A list of words is a sentence if the list is obtained by appending a list which is a noun phrase, a list which is a verb phrase, and a list whose single element is a period. Your program can be used to check if a given sentence can be generated by the grammar. An example interpreter session is also provided below.

   Grammar:
   sentence → noun-phrase verb-phrase
   noun-phrase → article noun
   article → a | the
   noun → manager | programmer | code
   verb-phrase → verb noun-phrase
   verb → writes | reviews

   Example:

```
| ?- sentence([the, manager, reviews, the, code]).
| true.
```

   (b) (5 pt) Can you use the same program to generate all possible sentences that can be derived from the grammar? If so, write the program.

   (c) (2 pt) Does the order of the subgoals in your rules make a difference?

Sol

```
sentence([]).
sentence([A,B|Tail]):- noun-phrase(A,B),checkVerbPhrase(Tail).
checkVerbPhrase([A,B,C|Tail]):- verb-phrase(A,B,C), isNull(Tail).


noun-phrase(A, B):- article(A),noun(B).
```

```
7  verb-phrase(A,B,C) :- verb(A),noun-phrase(B,C).
8  isNull([]).
9
10
11 article(a).
12 article(the).
13 noun(manager).
14 noun(programmer).
15 noun(code).
16 verb(writes).
17 verb(reviews).
```

(a) Yes. for this particular implementation type in `sentence([A,B,Tail]).` into prolog continue to ask for more solutions until no more backtracking can be done. It is not optimized so that there are no repeat answers.

(b) No.

4. (20 pt) Write a Prolog program for solving the prerequisite problem:

   (a) Write a Prolog program to represent the prerequisite relations for all the undergraduate courses (see 100, 200, 300 and 400 level courses here `http://catalog.iastate.edu/azcourses/com_s/`). Some courses have requirements on grades, you do not need to include these constraints. Also, you can assume that courses outside coms (e.g. math, stat, engl) have no prerequisite.

   (b) Write a query, ?-cantake(coms342, X), asking "if you have taken COM S342, what other courses you can take without a prerequisite waiver". List all the courses that satisfy the query and also have a prerequisite.

   (c) Write a query, ?-totake(coms342, X), asking "to take COM S342, what is the set of courses you need to take?". Return all possible sets.

   Sol. solution in problem4.pl