CSE341 – Programming Languages (Fall 2020)

Homework #5 (Replacement for Midterm)

Handed out: December 30, 2020.

Due: 11:55pm January 24, 2021.

Hand-in Policy: Source code and documentation should be submitted online as a single .zip or .rar file with naming convention studentid_lastname_firstname_hw5.zip via Moodle by the submission deadline. No late submissions will be accepted.
Collaboration Policy: No collaboration is permitted. Any cheating (copying someone else's work in any form) will result in a grade of -100 for the first offense and -200 for the subsequent attempts.
Grading: Each homework will be graded on the scale 100. Unless otherwise noted, the questions/parts will be weighed equal.

Consider the following further simplified version of Prolog we have discussed in class:

- Axioms as given in Horn clause form.
- Variables in head are universally quantified.
- Variables in body only are existentially quantified.
- A query is a clause without a head.
- A fact is a clause without a body.
- A prolog program is composed of a list of clauses and one or more queries.
- There are no functions. Note that lack of functions will render this language very weak since useful arithmetic and many other things cannot be done without functions.

A BNF for this simplified version Prolog is given below:

```
<program> ::= <clause list> <query> | <query>
<clause list> ::= <clause> | <clause list> <clause>
<clause> ::= <predicate> . | <predicate> :- <predicate list>.
<predicate list> ::= <predicate> | <predicate list> , <predicate>
<predicate> ::= <atom> | <atom> ( <term list> )
<term list> ::= <term> | <term list> , <term>
<term> ::= <numeral> | <atom> | <variable>
<query> ::= ?- <predicate list>.
<atom> ::= <small atom> | "<string>"
<small atom> ::= <lowercase letter> | <small atom> <character>
<variable> ::= <uppercase letter> | <variable> <character>
<lowercase letter> ::= a | b | c | … | x | y | z
<uppercase letter> ::= A | B | C | … | X | Y | Z
<numeral> ::= … integer numbers …
<character> ::= … all characters …
<string> ::= <character> | <string> <character>
```

Assume that someone has written a lexer and a parser that can take a program in this language and generate a data structure in Common Lisp.

- The outcome of the parser is a list of horn clauses.
- Each horn clause is a list of two entries, namely head and body.
- Head part is itself a list.
    - If the clause is a query, head will be nil.
    - Otherwise, it will be a predicate.
- Body is also a list with zero more entries.
    - If the clause is a fact, the body will be nil.
    - Otherwize, it will be a list of predicates.
- A predicate is a list with two entries: (predicate_name list_of_parameters)
    - The first entry is a string indicating the name of the predicate.
    - The second is a list of (possibly empty) parameters.
    - The list of parameters can have string or numeric entries. String entries starting with capital letters indicate that the parameter is a variable. String entries starting with lowercase letters indicate name of objects (NOTE: object names cannot start with caps). Numeric entries are treated as Common Lisp integers.

For example, the following code:

```
legs(X,2) :- mammal(X), arms(X,2).
legs(X,4) :- mammal(X), arms(X,0).
mammal(horse).
arms(horse,0).
?- legs(horse,4).
```

will generate the following list:

```
(
        (  (“legs” (“X” 2)) ( (“mammal” (“X”)) (“arms” (“X” 2)) ) )
        (  (“legs” (“X” 4)) ( (“mammal” (“X”)) (“arms” (“X” 0)) ) )
        (  (“mammal” (“horse”)) () )
        (  (“arms” (“horse” 0)) () )
        (  () (“legs” (“horse” 4)) )
)
```

Write a Common Lisp function such that when a parsed list of Horn clauses is given (as defined above) as input, it will answer all the queries in this list of clauses. You will use resolution and unification methods discussed in class to prove the queries and return the list of values for all variables for which the query is true. An empty list on return means that the query is false. Make sure that your query answer function halts.