Computer Science C24
University of Toronto Scarborough

Homework Exercise # 5

Handing in and marking

For all exercises/assignments in this course, you need to submit your solutions to the pencil-and-paper questions on crowdmark and your solutions to the programming questions on MarkUs. Your pencil-and-paper solutions will be marked with respect to correctness, clarity, brevity, and readability. Your code will be marked with respect to correctness, efficiency, program design and coding style, clarity, and readability. This exercise counts for 6% of the course grade.

In this exercise we represent propositional formulae (no quantifiers) as Prolog terms as follows:

- `tru` is a formula.
- `fls` is a formula.
- `variable(V)` is a formula iff `V` is a suitable identifier (in this question we will use "if `V` is an atom", which you can check in Prolog with `atom(V)`).
- `neg(F)` is a formula iff `F` is a formula.
- `and(FList)` is a formula iff every element in the list `FList` is a formula.
- `or(FList)` is a formula iff every element in the list `FList` is a formula.
- `implies(F0,F1)` is a formula iff both `F0` and `F1` are formulae.

1. Write a predicate `formula(?F)`, which succeeds iff `F` is a valid formula.

2. We can represent a truth assignment (an assignment of values to variables) by a Prolog list of the form `[Var1/Value1, Var2/Value2, ..., VarN/ValueN]`. Write a predicate `sub(?F,?Asst,?G)` which succeeds iff `G` is a formula which is a result of substituting the variables of `F` with corresponding values from the assignment `Asst`. You can assume that the truth assignment `Asst` is at least partially instantiated (i.e. the length of the list is known).

3. Write a predicate `eval(?F, ?Asst, ?V)` which succeeds iff the formula `F` has value `V` (either `tru` or `fls`) under the truth assignment `Asst`. You can assume that the truth assignment `Asst` is at least partially instantiated (i.e. the length of the list is known).

Assume all inputs are valid. In particular, an assignment of values to variables mentions each variable at most once. Please, consult the starter tester file `testProp.pl` for some examples of using these predicates. You may want to look at the documentation for `PLUnit` first http://www.swi-prolog.org/pldoc/package/plunit.html.