CSC592CM – Programming Assignment #2

Due 11/2/06
Version 1.0

Write a LISP/Scheme program that implements a simple forward-chaining rule-based system that performs some inferential task that interests you.

First write a MAKE-RULE procedure that creates rules and puts them on a list of *RULES*. Your system should have at least 10 rules in it. Use data abstraction by writing procedures such as GET-ANTECEDENTS to extract the antecedents from the rules.

In order to avoid having to write complicated matching and binding procedures, represent assertions as symbols, so that HAPPY is used in place of (X IS HAPPY). Then a rule can be something like (IF (rich healthy) THEN (happy)).

Your main procedure will be a DO loop (or recursive function in the case of Scheme) that repeatedly evaluates all the *RULES* you have created to see if the antecedents match working memory. If there is a match, then the consequents are added to working memory.

To see if the antecedents of the rule match working memory, all you have to do is to check whether the antecedents are a subset of working memory (if you are using Scheme you will need to write a function similar to the Common Lisp predicate SUBSETP). Firing a rule is then just a matter of appending the consequents of the rule to working memory.

Hand in a listing of your program as well as a program run that shows that your system is capable of chaining inferences, i.e. going from A and IF A THEN B and IF B THEN C to C.

Students are encouraged to discuss design and implementation strategies, but each student has to submit their own implementation, that is, the actual implementation and testing has to be your own and cannot be shared.