CS322 Fall 1997
Assignment 1

Due: 1:30pm, Friday 12 September 1997.

Question 1

Given the knowledge base:

$$a \leftarrow b \wedge c.$$
$$a \leftarrow e \wedge f.$$
$$b \leftarrow d.$$
$$b \leftarrow f \wedge h.$$
$$c \leftarrow e.$$
$$d \leftarrow h.$$
$$e.$$
$$f \leftarrow g.$$
$$g \leftarrow c.$$

where $\{a, b, c, d, e, f, g, h\}$ is the set of all atoms.

(a) [3 marks] Give a model of the knowledge base.

(b) [3 marks] Give an interpretation that is not a model of the knowledge base.

(c) [2 marks] Give two atoms that are logical consequences of the knowledge base.

(d) [2 marks] Give two atoms that are not logical consequences of the knowledge base.

Question 2

This question considers how an explicit semantics can be used to debug programs. The file

```
˜cs322/cilog/elect_bug.cil
```

contains an axiomatization of the electrical wiring domain (Figure 3.1, page 69, of the text), but it contains a buggy rule. The aim of this exercise is to use CILOG to automatically find the buggy rule. This is the rule that is false in the intended interpretation given in Figure 3.1, given the denotation of the symbols as on page 70 of the text. To find the buggy rule, you won't even need to look at the source code![1] All you need to know is the meaning of the symbols in the program, and what is true in the intended interpretation.

The following is a trace showing how to run CILOG on this example:

```
keats<8> cd ˜cs322/cilog
keats<9> ./cilog

CILOG Version 0.11. Copyright 1997, David Poole.
CILOG comes with absolutely no warranty.
All inputs end with a period. Type "help." for help.
cilog: load 'elect_bug.cil'.
CILOG theory elect_bug.cil loaded.
cilog: ask lit(L).
Answer: lit(l1).
  [ok,more,how,help]: more.
Answer: lit(l2).
  [ok,more,how,help]: more.
No more answers.
cilog:
```

The first answer, $lit(l1)$, is false in the intended interpretation. You can use the how questions of CILOG to find a clause whose head is false in the intended interpretation and whose body is true. This is a buggy rule.

Find the buggy rule in the file `˜cs322/cilog/elect_bug.cil`.

You need to hand in a trace of cilog that demonstrates how the how command can be used to debug programs. You need to highlight (e.g., by circling) the buggy rule found in your output. (You can get a trace by either cutting and pasting from a window, using "script" in Unix, from the emacs buffer, or any other mechanism you like).

[1] You can look at the source code to find the buggy rule if you like, but that won't help you in this exercise.