Knowledge-Based Systems
Exercise 1 (assessed)

Please hand in your solutions to this exercise at the lecture on Thursday 21st March 2002. Remember to hand in evidence that these programs work, in addition to the program listings.
This exercise is worth *half* of the total continuous assessment marks available for CM0377.

Question 1

Download the family relationships program that uses lists (new_fam1.pl) and extend it with **brother_of**, **sister_of** and **ancestor_of** rules. **NB** the **ancestor_of** rule should succeed if someone is a parent, grandparent, great-grandparent, great-great-grandparent, etc. In order to demonstrate that it works, you will need to increase the number of people known to the program.

Question 2

Write a Prolog program that has three types of clauses containing module information:

1. The title and lecturer associated with a module number (e.g. module number 42 has title 'Management Skills' and lecturer 'Jim Kirk')
2. For each person attending a given module, a clause associating the person's student number and the module number (e.g. the person with student number 143255 is attending module number 42; the person with student number 143255 is attending module number 55; the person with student number 666333 is attending module number 42, …)
3. For each student, his/her name, student number and his/her year of study, which will either be 1, 2 or 3. (E.g. 'Jane Jones' has student number 143255 and is in year 1; 'Bill Evans' has student number 14555 and is in year 3, …)

Your program should contain additional clauses such that the user can issue the following three kinds of goals:

1. print_year_of_study_for(<student number>)
   (Which should print out the year of study for the student number specified)
2. print_modules_for(<student name>)
   (Which should print out the name and number of each module (s)he is taking)
3. print_students_lectured_by(<lecturer name>)
   (Which should print out the names and student numbers of all students who are taking some module given by the named lecturer)

Additional credit will be given if your program behaves appropriately, e.g. by displaying *not found,* when the specified student number, student name or lecturer name cannot be found. To achieve this you will need to use the negation-as-failure (\+) operator introduced briefly in lecture 1.

Question 3

Consider the following propositional clausal logic program *P:*

```
confuses_people; gives_clear_information :- politician.
ambitious :- politician, confuses_people.
:- gives_clear_information.
politician.
```

(a) Write down the Herbrand Base of *P.*
(b) Write out all possible Herbrand Interpretations of *P,* and for each one determine whether or not it is a model of *P.*
(c) Hence show that the clause:
    `ambitious`
    is a logical consequence of *P.*
(d) Now prove *by refutation* (i.e. using resolution) that
    `ambitious`
    is a logical consequence of *P.*

Andrew Jones
26th February 2002