Coursework 2

What to do. Form a group of 2-4 people (larger groups and individual work are not allowed). Then select *one* of the problems below to work on. The choice of programming language is up to you. You will have to meet the following deadlines:

- Email me your group and chosen problem by **Monday, 4 March 2002, 4pm**.
- Attend a lab session in the following week (details to be announced) and convince me that (a) you're making progress and (b) every group member is contributing to the solution (that is, all group members need to be present).
- Submit your final solution by **Thursday, 21 March 2002, 6pm** (electronically).

Submission. An acceptable solution will (at least) consist of a (working) *program* (including the full source code) and a (very short) *report* describing how things work. Only programs that run on the Departmental Linux system are acceptable. Documents should be platform-independent (e.g. PDF, but not MS Word or alike). Maybe you want to take this opportunity to learn LaTeX. For your submission, *one* member of your group should place all the files you wish to submit in the following directory:

`~/cs3aur/submit/`

Problem 1 (Normal forms). Write a program that allows for the input of formulas in first-order logic and offers the following services:

- Transformation into CNF and DNF (for formulas without quantifiers).
- Transformation into Prenex Normal Form.
- Transformation into Skolem Normal Form.

Possible extension:

- Find out about other normal forms and implement them as well.

Problem 2 (Temporal constraints). Write a program that allows the user to enter a number of temporal constraints using Allen's interval relations and then checks whether the corresponding interval network is consistent or not. Example:

Input: $(a, b) : \{before, meets\}$, $(b, c) : \{before\}$, $(a, c) : \{during, overlaps\}$
Output: inconsistent

Possible extensions:

- Visualise the output of consistent interval networks.
- Allow propositional connectives (like $\neg$ and $\vee$) to combine constraints.
- Try to help the user to turn an inconsistent network into a consistent one.

Problem 3 (Description logics). Write a program to check the satisfiability of an ABox in the description logic $\mathcal{ALC}$.

Possible extension:

- Implement TBox unfolding as well.