Prolog - Programming in Logic

Dr Ian Lewis

Hello and welcome to the "Prolog - Programming in Logic" course! You've already spent time learning where the commas and asterisks go in a variety of *imperative* programming languages (e.g. Java, Javascript, pretty much every other commonly used language). With a bit of luck you've also learned that using a *declarative* language based on the rigorous foundations of the functional calculus (e.g. SML) is still highly capable but provides greater opportunities to reason with clarity about the expected outcome of the execution. Prolog, based upon the relational calculus, presents another declarative paradigm for program execution particularly suited to particular problems (e.g. those involving searching through linked relations seeking a valid solution for a query).

I've included a few brief notes here to explain how the course works. I can be contacted at ian.lewis@cl.cam.ac.uk if you have any questions.

The department webpage for this course is:
https://www.cl.cam.ac.uk/teaching/current/Prolog

The videos and the PDF's of the slide decks are accessible via Moodle, To see the Prolog course in context, select it from the Part IB courses listed here:
https://www.vle.cam.ac.uk/course/index.php?categoryid=24501

Note the Moodle videos include pauses with helpful 'checklist' questions, but as a 'backup' the same videos (without the checklist questions) is available via YouTube:
https://www.youtube.com/playlist?list=PLFgheI-d1nD47H06F236cp5cFEQE0o8nx

Course structure

This course consists of 8 timetabled 'lectures' starting on Monday 26th Feb 2024. Each lecture is supported by a video explaining the core *programming* content, and an in-person lecture will review that material providing a broader context and clarify any subtleties or implications. The objective is that those videos should be watched before the lectures, complimented by simple hands-on programming in the Prolog environment to explore/test any details, and we can spend less time buried in the minutiae of the Prolog language during the lectures.

With reference to the materials list on the Moodle course page: https://www.vle.cam.ac.uk/course/view.php?id=252621
Overall *per-lecture* on the Prolog course, we have the following. Please note the optional elements are unrelated to the assessment or marking of the course:

- ➢ One or more short (10..20min) videos explaining the core material (required). These videos may contain embedded multiple-choice questions to check your understanding (optional)
- ➢ One or more Quiz to be viewed anytime after viewing the video to check your understanding (optional)
- ➢ The PDF slide deck (and possibly a video) of the in-person lecture (optional).

On that Moodle course page linked above you will find links to videos for all the content with the particular lecture slot with which they are best associated. If preferred you can also review this content at your own pace although that may then be out-of-sync with the in-person lectures.

Supervisions

There are two supervisions associated with the course and the "Recommended supervision work" is included on the Moodle course page linked above. Your supervisor might choose to set you something different.

Practical Work

For most of you, Prolog will be a new programming language using a new programming paradigm. The absolute #1 mistake to avoid early on in the course is to believe *all* programming languages are imperative and hence spend time trying to work out how to implement corresponding simple imperative features in your Prolog programs.

Familiarity with Prolog is most easily attained by writing a few simple programs while avoiding the #1 mistake listed above. SWI Prolog (https://www.swi-prolog.org/) is a mature implementation and I recommend you install that and experiment with it as early as is practicable. The 'hello world' of Prolog is to define a few relations of who is the child of who, and then create a rule that declares the meaning of a 'grandchild'.

Keep in mind that Prolog is designed for searching algorithms over data stored in the form of relations, and to do that does not require much of the real-world API support normal in common application languages. So if you find yourself trying to interface Prolog to the latest Windows .Net API or read data from a network socket, stop.

Interaction in the Lectures

Feel free to bring up questions during the lectures, and there will usually be time at the end of the lectures for people to stop by the front desk and compare notes or ask questions then. Whatever question you ask, you can be sure that plenty of other people in the room will benefit from the follow-up.