"Prolog Parentage"

Programming Lab 4

Concepts of Programming Languages
CSCI305, Spring 2013

Due: April 26, 2013 at 11:59 pm

Prolog

For this lab, you will use Prolog. Download Prolog from here: `http://www.swi-prolog.org/download/stable`. Prolog is intalled in the departmental lab on Linux.

Dataset

Begin by downloading this file (`http://nisl.cs.montana.edu/~pdonnelly/CSCI305/docs/royal.pl`).[1] This file is a set of Prolog facts that represents the genealogy of the royal House of Windsor.

Getting Started

Next, create a new file `[lastname]_[firstname].lab4.pl`.[2] This file should be in the same directory as `royal.pl`. In this file, you will include all your Prolog rules you will be asked to write in the lab.

Your program should begin with your name as a comment and will load the file `royal.pl`.

```
% First_name Last_name
% CSCI 305 Lab 4
consult('royal.pl')
```

[1]This data was adapted from `http://ftp.aset.psu.edu/~saw/royal/`

[2]Note that `.pl` is an extension also used by Perl and may confuse syntax highlighting programs.

Warmup

Check that everything is working. First load your file in SWI-Prolog by choosing File→Consult and choosing the file your created above. Next, run the query:

```
?- parent(X, 'Queen Elizabeth II').
```

The symbol `?-` is the Prolog query prompt. You do not need to retype this. This query asks who is the parent of Queen Elizabeth II. Prolog first returns the result `X = 'King George VI'`. Type a semicolon (;); this will prompt Prolog for the next result. Prolog then returns `X = 'Lady Elizabeth Bowes-Lyon'`. The period following the second result indicates there are no additional results.

The entire query and result interaction should appear as:

```
?- parent(X, 'Queen Elizabeth II').
X = 'King George VI' ;
X = 'Lady Elizabeth Bowes-Lyon'.
```

Replicate this exchange on your machine.

Parents

In this lab you will create a number of Prolog rules that define a number of common familial relationships.

In Prolog parlance, we define a rule as its name and its arity. For instance the rule `mother/2` indicates a rule named `mother` with two arguments.

```
mother(M,C):- parent(M,C), female(M).
```

Add this rule to your Prolog file. Reload your file. Test your new rule with the query:

```
?- mother(X, 'Queen Elizabeth II').
```

which should return the single result:

```
X = 'Lady Elizabeth Bowes-Lyon'.
```

Now write the rule `father/2` and add it to your file.

More Rules

Now write a set of additional rules. In your rules, you are encouraged to make use of other rules you have written. This may effect the order you choose to implement these rules.

- `spouse/2`
- `child/2`
- `son/2`
- `daughter/2`
- `sibling/2`
- `brother/2`
- `sister/2`
- `uncle/2` - two rules: one by blood, one by marriage.
- `aunt/2` - two rules: one by blood, one by marriage.
- `grandparent/2`
- `grandfather/2`
- `grandmother/2`
- `grandchild/2`

For the next two rules, make use of rules you have already written. It may take more than one rule to define these functions. You may find tracing useful. To debug, you can use the keyword `trace` to enable tracing and `notrace` to disable.

Write the rules:

- `ancestor/2`
- `descendant/2`

Numbers in Prolog

Now we will write some Prolog rules that require numeric comparisons. Write the rules:

- `older/2`
- `younger/2`

The rule `older(X, Y)` indicates person X is older than person Y. The rule `younger/2` should be written in a similar manner.

Lastly, write the rule `regentWhenBorn/2`. This rule, `regentWhenBorn(X, Y)`, should ask who was King or Queen (X) when person Y was born.

Extra Credit (10 pts)

Write rule(s) that define the first cousin relationship, `cousin/2`. You may need to look up the formal definition of cousin. You do not need to support second or third cousins, nor cousins removed.

Question EC1: What is result of query `?- cousin(X, 'Prince Charles').`?

Question EC2: What is result of query `?- cousin('Prince Charles', X).`?

Enable tracing and run both of these queries again.

Question EC3: Which resulted in more steps? Explain why. Provide both of these traces in a file `[lastname]_[firstname].lab4.ec.trace.txt`.

Lab Questions

For these lab questions, you are expected to use Prolog to answer. Answers that are given without providing the Prolog rules that derive it will receive no credit. Some queries have multiple answers and you are expected to provide all results to the lab questions. Use the semicolon (;) to prompt Prolog for additional results.

Question 1: What is result of query `?- father(X, 'Queen Elizabeth II').`?

Question 2: What is result of query `?- grandmother(X, 'Queen Elizabeth II').`?

Question 3: What is result of query `?- grandfather(X, 'Queen Elizabeth II').`?

Question 4: What is result of query `?- grandparent(X, 'Queen Elizabeth II').`?

Question 5: What is result of query `?- grandparent('Queen Elizabeth II', X).`?

Question 6: What is result of query `?- sibling(X, 'Queen Elizabeth II').`?

Question 7: What is result of query `?- son(X, 'Queen Elizabeth II').`?

Question 8: What is result of query `?- daughter(X, 'Queen Elizabeth II').`?

Question 9: What is result of query `?- aunt(X, 'Lady Diana Spencer').`?

Question 10: What is result of query `?- spouse(X, 'Prince William').`?

Question 11: What is result of query `?- ancestor(X, 'Prince Henry').`?

Question 12: What is result of query `?- descendant('Queen Victoria', X).`?

Question 13: What is result of query
`?- older( 'Prince Henry','Prince William').`?

Question 14: What is result of query `?- older(X, 'Queen Elizabeth II').`?

Question 15: What is result of query
`?- regentWhenBorn(X, 'Queen Elizabeth II').`?

Troubleshooting

This lab requires an independent study of the Prolog language. You are encouraged to use any web tutorials and resources to learn Prolog. Given the size of the class, I will not be able to debug your code for you. Please do not send panicked emails requesting I fix your bug for you. Allow yourself plenty of time, and use patience, perseverance, and the internet to debug your code. I will gladly answer clarifying questions about the goals and instructions of the Lab assignment.

Lab Questions

The following questions are for feedback and evaluation purposes. Points are awarded for any sincere answer.

Question 16: Name something you like about Prolog. Explain.

Question 17: Name something you dislike about Prolog. Explain.

Question 18: Did you enjoy this lab? Which aspects did you like and/or dislike?

Question 19: Approximately how many hours did you spend on this lab?

Question 20: Do you think you will use Prolog again? For which type(s) of project(s)?

Submission

Each student will complete and submit this assignment individually. Do not consult with others. However, you are encouraged to use the internet to learn Prolog but not to research the questions asked in this lab.

Comment your program heavily. Intelligent comments and a clean, readable formatting of your code account for 20% of your grade.

Save the final version of your program as `[lastname]_[firstname].lab4.pl`. Type your lab questions in plain text as `[lastname]_[firstname].lab4.txt`. Include your name in the text file.

You will email these files to `msucsci305@gmail.com`. Use the subject line `[lastname]_[firstname] Lab4 Submission`. If you completed the extra credit, also submit your trace file described above. Do not archive your files but instead use separate attachments. Email your files before 11:59pm on the due date. Late submissions will not be accepted.