LAB1

Welcome to the first lab!

The purpose of this lab is to get you up and running with the "SWI Prolog" toolchain and the "Learn Prolog Now!" tutorial (which are both well-supported free resources that form the core of the material in the first part of the unit. This week begins with a simple subset of Prolog known as Datalog which allows you to solve queries over simple facts and rules.

The three main objectives of this lab are to

* Install on your own computer (a free stable release copy of) SWI-Prolog
* Begin working through chapters 1-3 of the excellent "Learn Prolog Now!" tutorial
* Begin working through the movie database example on the SWI webservice

**Part 0 (Prolog installation)**

Use the [Download SWI-Prolog](https://www.swi-prolog.org/Download.html) and [Get Started](https://www.swi-prolog.org/pldoc/man?section=quickstart) links to install the Prolog engine "SWIPL"

Although SWI installation is straightforward on all platforms you are advised to

* Take a sneak peak at this week's SWI toolchain slides (which contain some useful instsllation and running tips)
* Visit the [SWI Prolog](https://www.swi-prolog.org/) homepage and start to have a very quick look through the [Documentation](https://www.swi-prolog.org/pldoc/doc_for?object=manual)
* Feel free to work together with colleagues sitting near you in the lab
* If all that fails then raise your hand to seek help from unit staff

If you cannot complete installation during the lab then use the instructions in the toolchain slides to do one of the following

* Use a preinstalled copy of SWI on one of the lab machines
* Use the SWISH webservice to complete the other tasks

**Part 1 (Basic syntax)**

Read through [chapter 1](http://www.learnprolognow.org/lpnpage.php?pagetype=html&pageid=lpn-htmlch1) of the free online version of LPN

Copy the clauses of each example knowlede base into a text file so that you can use SWIPL to run the relevant queries

Solve the following exercises (which have been taken from Section 1.3 and supplemented with answers for the first part of each exercise):

* **Exercise 1.1** Which of the following sequences of characters are atoms, which are variables, and which are neither?
  + vINCENT **Answer**:atom
  + Footmassage
  + variable23
  + Variable2000
  + big\_kahuna\_burger
  + 'big kahuna burger'
  + big kahuna burger
  + 'Jules'
  + \_Jules
  + '\_Jules'
* **Exercise 1.2** Which of the following sequences of characters are atoms, which are variables, which are complex terms, and which are not terms at all? Give the functor and arity of each complex term.
  + loves(Vincent,mia) **Answer**:complex term (loves/2)
  + 'loves(Vincent,mia)'
  + Butch(boxer)
  + boxer(Butch)
  + and(big(burger),kahuna(burger))
  + and(big(X),kahuna(X))
  + \_and(big(X),kahuna(X))
  + (Butch kills Vincent)
  + kills(Butch Vincent)
  + kills(Butch,Vincent
* **Exercise 1.3** How many facts, rules, clauses, and predicates are there in the following knowledge base? What are the heads of the rules, and what are the goals they contain?
  + woman(vincent). **Answer**:fact
  + woman(mia).
  + man(jules).
  + person(X):- man(X); woman(X).
  + loves(X,Y):- father(X,Y).
  + father(Y,Z):- man(Y), son(Z,Y).
  + father(Y,Z):- man(Y), daughter(Z,Y).
* **Exercise 1.4** Represent the following in Prolog:
  + Butch is a killer. **Answer**:killer(butch).
  + Mia and Marsellus are married.
  + Zed is dead.
  + Marsellus kills everyone who gives Mia a footmassage.
  + Mia loves everyone who is a good dancer.
  + Jules eats anything that is nutritious or tasty.
* **Exercise 1.5** Suppose we are working with the following knowledge base:
  + wizard(ron).
  + hasWand(harry).
  + quidditchPlayer(harry).
  + wizard(X):- hasBroom(X), hasWand(X).
  + hasBroom(X):- quidditchPlayer(X).

How does Prolog respond to the following queries?

* + wizard(ron). **Answer**:true
  + witch(ron).
  + wizard(hermione).
  + witch(hermione).
  + wizard(harry).
  + wizard(Y).
  + witch(Y).

**Part 2 (Basic queries)**

Download the SWISH [movies.pl](https://swish.swi-prolog.org/example/movies.pl) example program and (either working in your browser or working with a local version of SWIPL using File->Download to save a copy of the prolog source), write queries to answer the following questions:

* In which year was the movie American Beauty released? **Answer**: ?-movie(american\_beauty,Y).
* Find the movies released in the year 2000.
* Find the movies released before 2000.
* Find the movies released after 1990.
* Find an actor who has appeared in more than one movie.
* Find a director of a movie in which Scarlett Johansson appeared.
* Find an actor who has also directed a movie.
* Find an actor or actress who has also directed a movie.
* Find the movie in which John Goodman and Jeff Bridges were co-stars.

**HINT**: the answers to this part are in the examples queries contained in the file itself!

**Part 3 (Basic rules)**

Now add to the knowledge base definitions of predicates with the following syntax and semantics:

|  |  |
| --- | --- |
| **Predicate Signature** | **Intended Meaning** |
| released\_after(M,Y) | movie M was released after year Y |
| released\_before(M,Y) | movie M was released before year Y |
| same\_year(M1,M2) | movies M1 and M2 were released in the same year |
| co\_star(A1,A2) | the actors/actresses A1 and A2 played in the same movie |

**HINT**: released\_after(M,Y) :- movie(M,Y2), Y<Y2.

**Part 4 (More complex rules)**

Now further add to the knowledge base definitions of predicates with the following syntax and semantics:

|  |  |
| --- | --- |
| **Predicate Signature** | **Intended Meaning** |
| plays(M,A,R) | A is an actor or actress who played a role R in the movie M |
| co\_direct(D1,D2) | D1 and D2 have co-directed some movie together |
| self\_direct(M,D) | D played in a movie M which they also directed |
| cameo(M,A) | A played them self in movie M |
| solo(M,A) | A was the only person who played in movie M |
| multi\_role(M,A) | A played two different roles in movie M |
| veteran(A) | A played in two movies more than 25 years apart |

**Part 5 (Fun Edge Cases)**

Add the following (hypothetical) facts to the knowledge base:

* movie(covid19, 2019).
* director(covid19, oliver\_ray).
* actor(covid19, nirav\_ajmeri, the\_hero).
* actor(covid19, nirav\_ajmeri, the\_villain).
* movie('2nd\_wave', 2020).
* director('2nd\_wave', oliver\_ray).

Try running some example queries (using the new predicates you defined above) to see if these (hypothetical) movies are atypical in any sense.