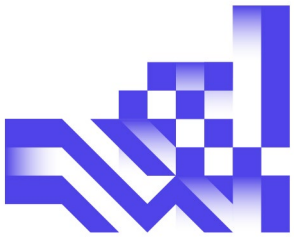
Hogeschool van Amsterdam

# Discrete Mathematics

*Practical*
*Assignment 1*



## Table of Contents

1 Assignment ....3
2 Environment set up ....3
3 Resources ....6
4 Delivery and grading ....7



# 1 Assignment

In this assignment, you will write your own expert system in Prolog.

Wikipedia quote:

*"In artificial intelligence, an **expert system** is a computer system that emulates the decision-making ability of a human expert.[1] Expert systems are designed to solve complex problems by reasoning through bodies of knowledge, represented mainly as if–then rules rather than through conventional procedural code.[2] The first expert systems were created in the 1970s and then proliferated in the 1980s.[3] Expert systems were among the first truly successful forms of artificial intelligence (AI) software.[4][5][6][7][8]*

*An expert system is divided into two subsystems: the inference engine and the knowledge base. The knowledge base represents facts and rules. The inference engine applies the rules to the known facts to deduce new facts. Inference engines can also include an explanation and debugging abilities.[9]* "

Your expert system can cover a freely chosen domain, but not birds and pets. Your program can be similar to *Tweety*[1]. To get a higher grade, you are encouraged to extend the knowledge base and the inference rules (see: section Delivery and grading).

The assignment consists of two parts: The first part is building an expert system in Prolog, containing facts, rules and examples of queries. The second part is building a manual with instructions to test your expert system.

More information about the requirements can be found in the section "*Delivery and grading*".

# 2 Environment set up

## 2.1 Local copy versus online version

Prolog is a logic programming language with its roots in Artificial Intelligence and Computational Linguistics. SWI-Prolog provides rich libraries for web programming, RDF handling, database access, networking, etc. We cannot expose the full system in the web version for various reasons, security being the most important. If you want to **work on a real application**, download your own copy of SWI-Prolog.

You can use the SWI Prolog environment which has all the features of Prolog, but we prefer you use the online version which will be sufficient for this assignment:

http://swish.swi-prolog.org/

[1] Tweety is an example of a small expert system, available in SWISH, the online version of SWI Prolog. You can find it through the main menu "*Examples* > *Example Programs* > *Classics* > *Expert System*".



SWISH is a limited subset of SWI-Prolog and can be used online. The online version provides you with some advantages:

- No installation required. Only needs a recent browser.
- You can play with some examples to get a basic feeling of what Prolog programming looks like.
- You can do exercises from Learn Prolog Now!
- You can **save your code** and **share** both the code and example queries with anyone on the web.

## 2.2 Usage

As mentioned earlier, an expert system is composed of a knowledge base and an inference engine. Your task is to build rules and facts and provide enough instructions to test your application. To introduce you to the development environment, let's use the following simple schematic knowledge base about cars:

```
haswheels(X) :- car(X);bike(X).
car(civic).
car(corolla).
bike(ninja).
```

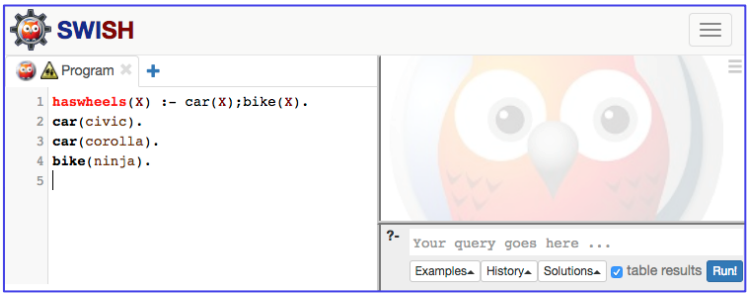
Four statements compose this small knowledge base. The first statement declares the *rule haswheels*. This rule could be read as "*X has wheels if X is a car or a bike*". The remaining lines, declaring car and bikes, are called *facts*.

You can load this knowledge base using two different strategies: online version or local copy.

### *2.2.1 Online version*

*Creating the knowledge base:*

- Go to https://swish.swi-prolog.org to access the online application. The editor will open.
- The editor allows many tabs. In the standard tab, create a program, by selecting the option "*Create program*".
- Paste the rules and facts into the program section. At this moment, your knowledge base is available for queries.



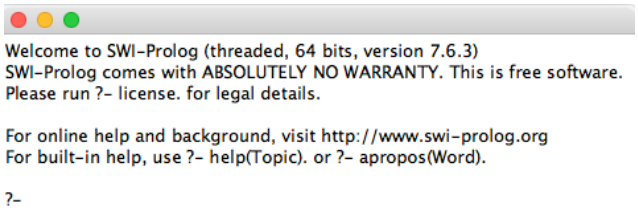
At any time, you can save your source-code in the cloud by using the menu option “File > Save”, or download to your local machine, using the “Download” option

*Querying your knowledge base*

You will see the query area on the right-bottom screen. The following screenshot shows an example of query and result, asking if a Corolla has wheels.


### 2.2.2 *Local copy*

We assume that you have installed SWI Prolog on your computer. If not, you can download it here. Load the SWI Prolog client from your local machine. The console will open, as illustrated below:




Use your preferred editor to save rules and facts. Do not forget to save your file with the .pl extension.

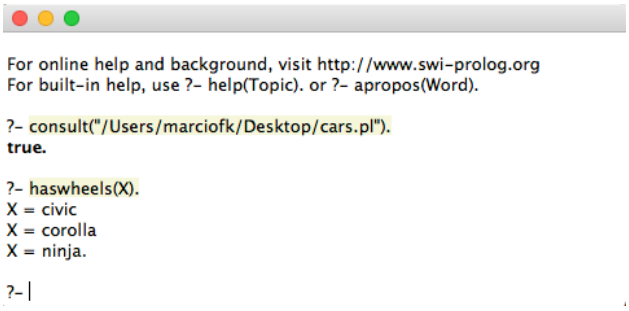
Back to the SWI console, use the *consult* function to load your knowledge base. Call the function specifying the full path of the file that contains your facts/rules base. For example:

*consult(“/Users/foo/Documents/cars.pl”).*

The function returns true if the syntax of the file is valid.

Now you can formulate your queries using the console. The example below asks for all elements (X) with wheels.






Now you know the basics of the development environment.

During the development of your exercise, make a clear distinction between what is the knowledge base and the set of query examples used to verify your application. It will help you when delivering the application for evaluation. For more information see the section “Delivery”.

## 3 Resources

**Text Book: Discrete Mathematics with Applications, International Edition**
Page 127 contains a brief introduction to Prolog and some examples of facts, rules and queries.

**http://www.learnprolognow.org/**
This site contains an introductory course to programming in Prolog. Read chapter 1 to be familiar with the development environment.

**http://swish.swi-prolog.org/example/expert_system.pl**
You will be redirected to the SWI Prolog online environment. The source-code presents a tiny expert-system for birds and pets.

**https://gist.github.com/adrianomelo/207c4da2f50744f04c9d**
An example of expert system

**http://www.ling.helsinki.fi/kit/2003k/ctl272/expertsystem.html**
An example of expert system



# 4 Delivery and grading

## 4.1 Delivery

You hand in your deliver work in two separate files.

- The source-code of your Expert System with the name *pa1.pl.txt.*
- A manual in the PDF format including information about your expert system and instructions for testing (*manual.pdf*)

Zip files will not be accepted. The submission of the files should occur separately. The two files should be handed in by the same student. See: all the rules regarding practical assignments in the Study Manual (section “Grading practical assignments and rules”).

## 4.2 Grading

See below the grading rubric for this assignment.

| **Passing grade (5.5)**<br>**Minimal requirements** | **+1**<br>**Organization** | **+ 3.5**<br>**Extra** |
|---|---|---|
| The program is based on the Tweety expert system of Swish Prolog but for a different domain.<br><br>The four unique queries (using different rules) give the expected results. These unique queries are asking at least four separate questions.<br><br>The program contains minimal two different rules of inference.<br><br>Every query runs without errors.<br><br>The predicates have logical names.<br><br>Manual: Explanation how each rule of inference works and rationale, and the explanation refers to the actual source code.<br><br>Manual: detailed Instructions for querying the expert system are provided in the manual. | Two files are delivered separately: manual.pdf and pa1.pl.txt (text version of the expert system).<br><br>Comments in the code are useful and helpful per rule of inference. | The manual contains one or more diagrams to help to understand how the expert system works.<br><br>The program is entirely different from the Tweety example. More specifically, the code does not have the set of *prove rules*, present in the example code.<br><br>More than two inference rules are present.<br><br>More than four queries are present, and the rationale per query is described.<br><br>All the extra features are documented in the manual. |