Lab 4: Prolog for Windows[1]

Practice this lab on your own. This lab should get you started using SWI-Prolog under MS-Windows.

1. Starting Prolog.

1.1. Implementations.

Prolog implementations are freely available from Prolog websites
http://www.swi-prolog.org or http://gnu-prolog.inria.fr
(both under Windows and Linux).These websites provide also links to manuals, books, online tutorials, and other related Prolog materials.

1.2. Starting Prolog and loading a program.

You can start it by clicking the Prolog icon. As a result, a SWI-Prolog console will open (see Figure 1).

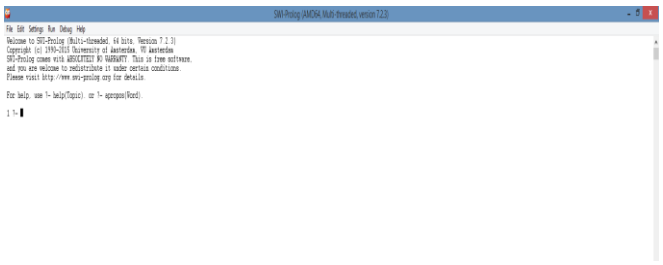

Figure 1.

Alternatively, you can start Prolog by opening a .pl file holding Prolog program text.

2. Using Prolog.

2.1. Menu commands.

The SWI-Prolog console has a menu for accessing the most commonly used commands. (see the first line in Figure 1). In particular, if you click on the button File of the menu you will see the following entries:

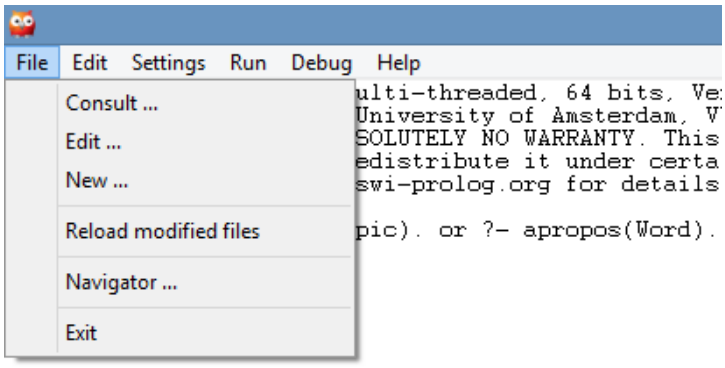

Figure 2.

[1] Acknowledgment: Dr. Alla Detinko, NUI Galway.

We will consider the first three commands in the next section. The command Reload modified files is to reload all loaded source-files that have been modified using the make command (section 2.2).
The command Navigator opens an explorer-like view on Prolog files and the predicates they contain: to view the predicates in a file you should find the file by Navigator and then click on the file name. Also Navigator provides an option to edit the selected file.
The menu button Settings allows you to change the font of the console as well as providing some additional options.
The button Run provides two options: Interrupt and New thread. Interrupt is to interrupt the running Prolog process. New thread creates a new window running in a separate thread of execution.
The button Help provides access to the online Prolog manual. In particular, it contains links to pages of the SWI-Prolog website.

2.2. Some useful commands.

This section contains a brief overview of important Prolog commands.

consult(+File)

Load a source-file from the current folder. The file-extension pl can be omitted (see Figure 3)

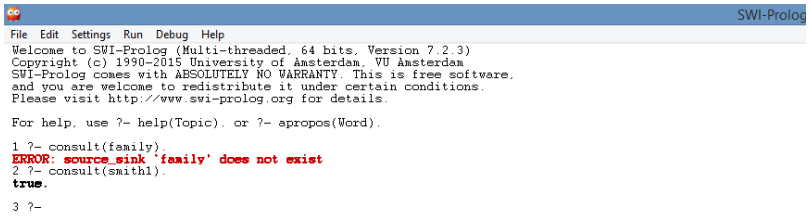

Figure 3

edit (+File)

Edit file with the given name. Also available from the menu (see section 2.1). If Prolog is started by opening a .pl file then the file name in the command edit can be omitted.

make

Reload all files that have been changed since they were last loaded. This command is normally used after editing one or more files.

Control-C

Try to interrupt the running Prolog process (see also Run in the menu). After execution, you will see the following line in the console:

Action (h for help) ?

If you really want to exit Prolog, type e. Else you can choose one of the following options.

a: abort b: break
c: continue e: exit

g: goals t: trace
h (?): help

3. Example.

Figure 4 shows the family tree of Smith's family.

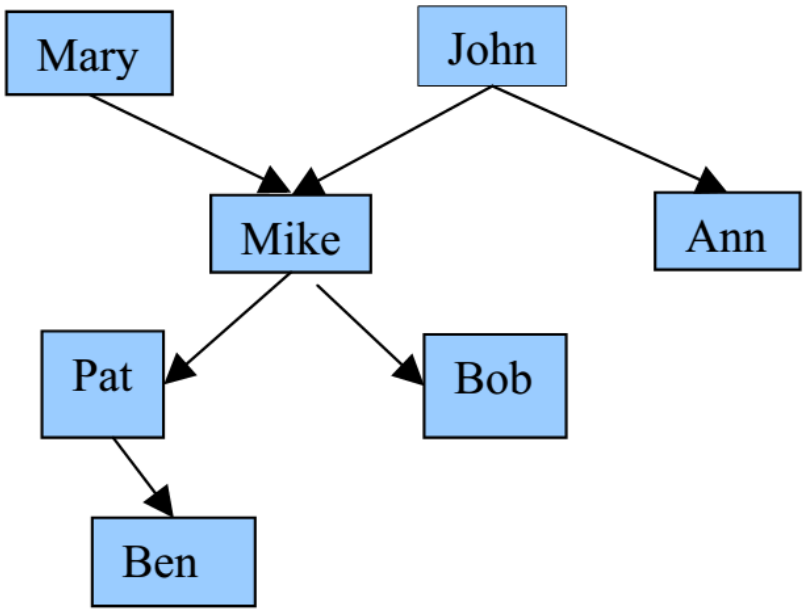

Figure 4.

To write a program which defines a family relation click the button File of the menu and choose the option New. Let us call the new file smith.pl. Type the text in the file and save it (see Figure 5).

```
parent(john, ann).      /*the fact that John is a parent of Ann*/
parent(john, mike).     %the fact that John is a parent of Mike
parent(mary, mike).
parent(mike, pat).
parent(mike, bob).
parent(pat, ben).
```

Figure 5

Here parent is the name of a family relation.
Note that the text in /* */ is a comment as well as the text between the percent character % and the end of the line (see the first and, respectively, the second lines in Figure 5).
Consult the file (see section 2.2); for example, use the command
?- consult(smith).
Now we can pose Prolog questions about the relation parent (see Figure 6).

```
?- parent(john, mike).          /* Is John a parent of Mike? */

Yes
?- parent(mike, ann).           /* Is Mike a parent of Ann? */

No
?- parent(mike, X).            / *Who is a child of Mike? */

X = pat

Yes
?- parent(mike, _).          /* Is Mike a parent? */

Yes
```

Figure 6

If we want to get a list of all pairs parent-child, type parent(X, Y), and then press repeatedly semicolon (;) (see Figure 7). To abort the process, press return.

```
?- parent(X, Y).
X = john
Y = ann ;

X = john
Y = mike ;

X = mary
Y = mike ;

X = mike
Y = pat ;

X = mike
Y = bob ;

X = pat
Y = ben ;
No                                  /*indicates that the list is full*/
```

Figure 7

To extend the program by adding new facts or rules, edit the file smith.pl. You can use, for example, option File/Edit of the menu, which will open the file smith.pl for editing.
For example, let us add information on the gender and define predicates mother, father (see Figure 8).

```
parent(john, ann).                          %the fact that John is a parent of Ann
parent(john, mike).                         %the fact that John is a parent of Mike
parent(mary, mike).
parent(mike, pat).
parent(mike, bob).
parent(pat, ben).

female(mary).                               %the fact that Mary is a female
female(ann).
female(pat).
male(john).                                 %the fact that John is a male
male(mike).
male(bob).
male(ben).

mother(X,Y) :- parent(X, Y), female(X).     %definition of the predicate mother
father(X,Y) :- parent(X, Y), male(X).       %definition of the predicate father
```

Figure 8

Consult the file smith.pl. Now you can ask new questions. For example, try the following:

```
?- female(X).
?- mother(X, mike).
?- mother(pat, Y).
?- father(X, Y).
```