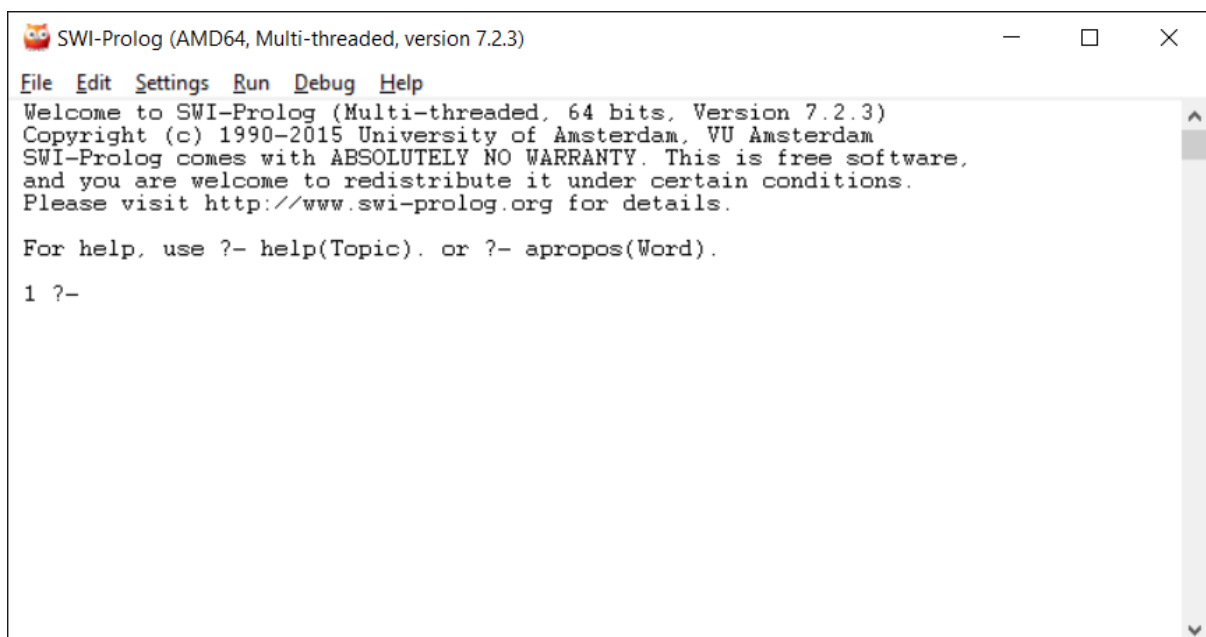


Lab 1

Use a text editor to write the following prolog program and save it as [likes.pl](#) in some folder where you intend to keep your prolog code. Then start SWI-Prolog and select the menu **File | Consult** to load in the program you have just written.

```
man(jim).
man(mary).
mortal(X) :- man(X).
likes(X,A) :- man(X), dog(A).

dog(rex).
dog(lassie).
```



In the Prolog interpreter try the following:

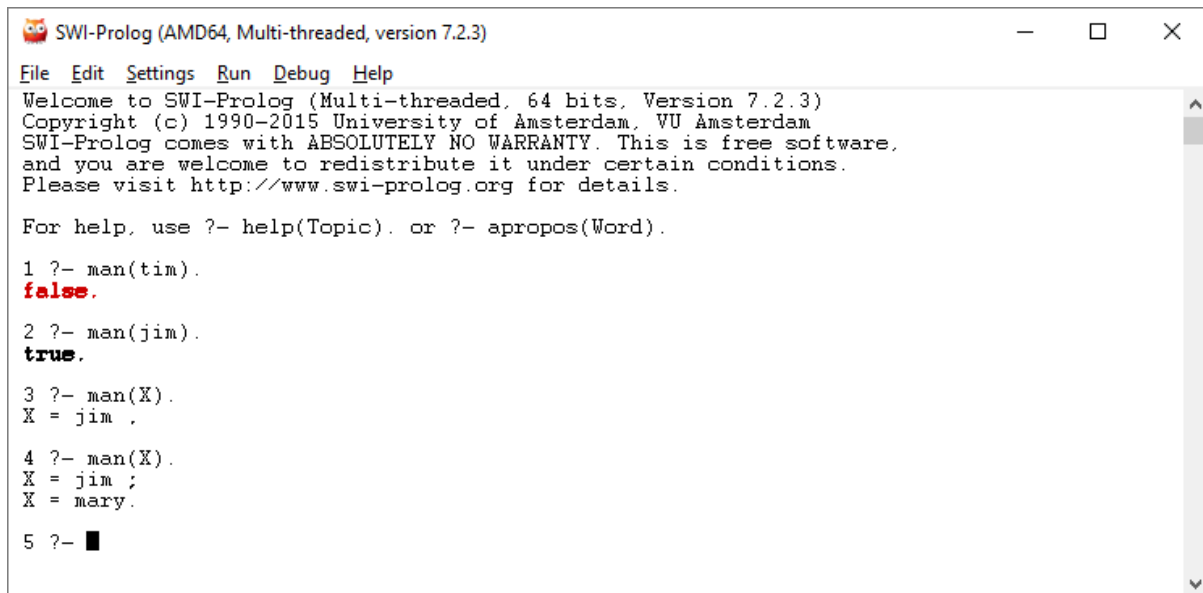
?- man(tim).

?- man(jim).

?- man(X). (and press enter after first value for X)

?- man(X). (and press ; after first value for X, should show more than one option for X)

These interactions will look like:

A screenshot of the SWI-Prolog (AMD64, Multi-threaded, version 7.2.3) window. The window has a title bar with the text "SWI-Prolog (AMD64, Multi-threaded, version 7.2.3)" and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with "File", "Edit", "Settings", "Run", "Debug", and "Help". The main text area contains the following text:

```
Welcome to SWI-Prolog (Multi-threaded, 64 bits, Version 7.2.3)
Copyright (c) 1990-2015 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?- man(tim).
false.

2 ?- man(jim).
true.

3 ?- man(X).
X = jim .

4 ?- man(X).
X = jim ;
X = mary.

5 ?- █
```

One assertions which exist within the Prolog database are deemed to be true, otherwise they are false. Referred to as the “closed world view”.

Next try some assertions which involve the 2 rules (Horn clauses):

?- mortal(sue).

?- mortal(mary).

?- mortal(N).

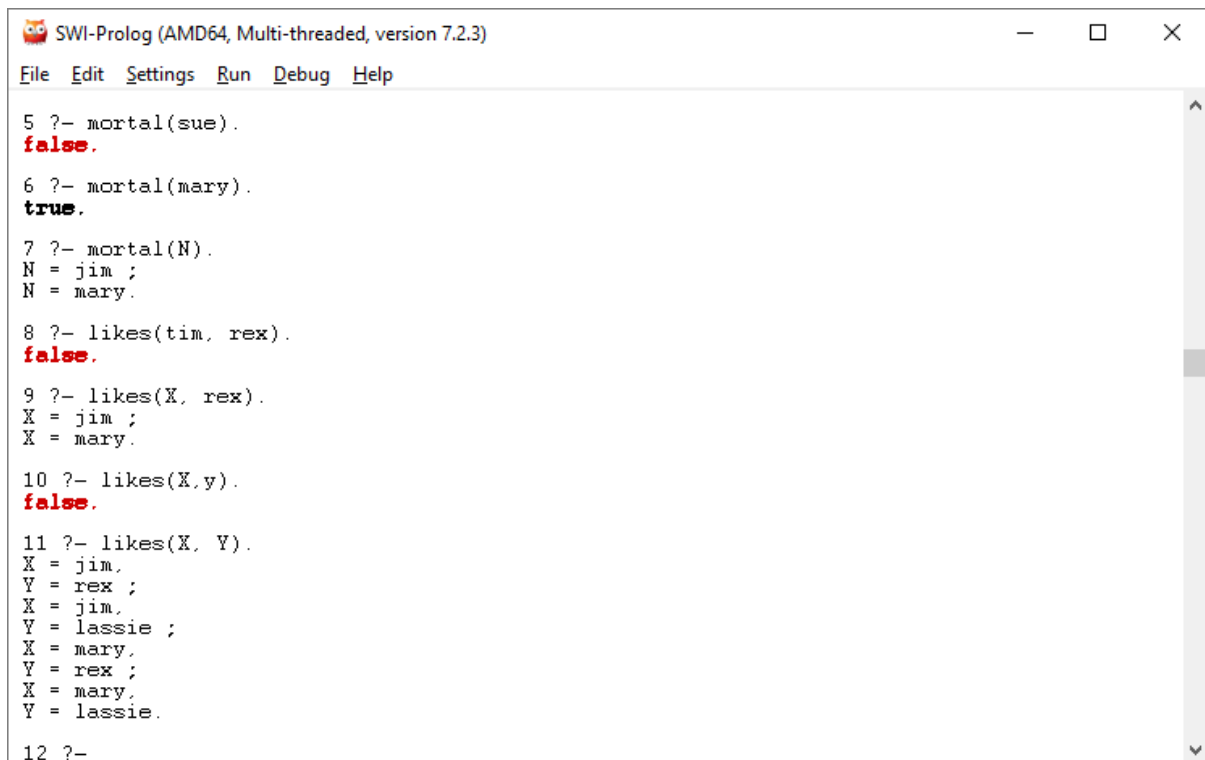
?- likes(tim, rex).

?- likes (X, rex).

?- likes(X, y).

?- likes(X,Y).

These interactions will look like:



```
SWI-Prolog (AMD64, Multi-threaded, version 7.2.3)
File Edit Settings Run Debug Help

5 ?- mortal(sue).
false.

6 ?- mortal(mary).
true.

7 ?- mortal(N).
N = jim ;
N = mary.

8 ?- likes(tim, rex).
false.

9 ?- likes(X, rex).
X = jim ;
X = mary.

10 ?- likes(X,y).
false.

11 ?- likes(X, Y).
X = jim,
Y = rex ;
X = jim,
Y = lassie ;
X = mary,
Y = rex ;
X = mary,
Y = lassie.

12 ?-
```

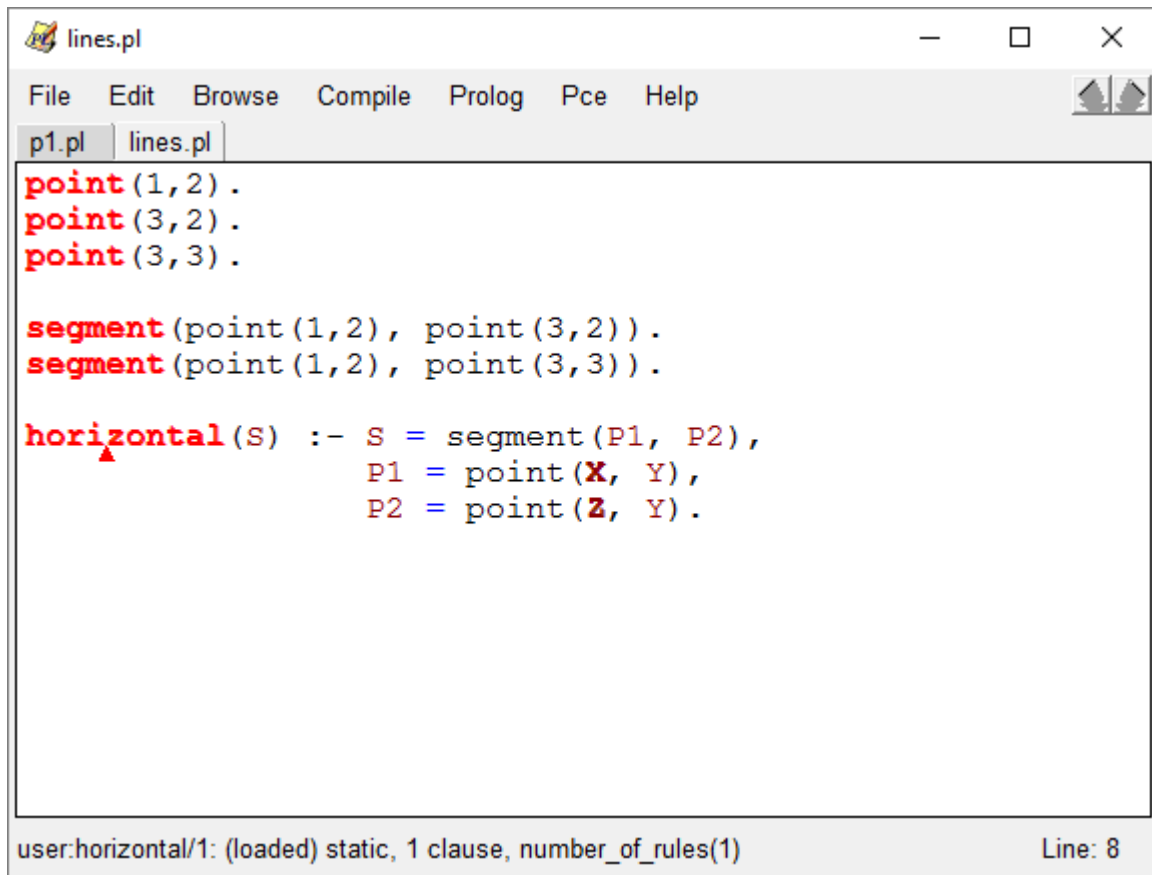
Exercises

Ex 1

Express in prolog the assertion that if X is a cat and Y is a dog, then X hates Y and Y chases X. Create a few cat facts and test this knowledge rule.

Ex 2

Write the following code and save it as [lines.pl](#) and then load it into SWI-Prolog.



```
lines.pl
File Edit Browse Compile Prolog Pce Help
p1.pl lines.pl
point(1,2) .
point(3,2) .
point(3,3) .

segment(point(1,2), point(3,2)) .
segment(point(1,2), point(3,3)) .

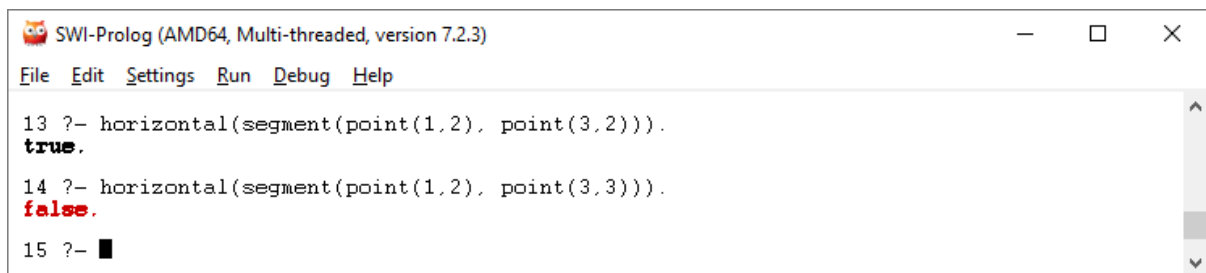
horizontal(S) :- S = segment(P1, P2),
                 P1 = point(X, Y),
                 P2 = point(Z, Y) .

user.horizontal/1: (loaded) static, 1 clause, number_of_rules(1)
Line: 8
```

Then try:

?- horizontal(segment(point(1,2), point(3,2))).

?- horizontal(segment(point(1,2), point(3,3))).



```
SWI-Prolog (AMD64, Multi-threaded, version 7.2.3)
File Edit Settings Run Debug Help
13 ?- horizontal(segment(point(1,2), point(3,2))).
true.
14 ?- horizontal(segment(point(1,2), point(3,3))).
false.
15 ?- █
```

Ex 3

Write a rule which decides if a line segment is vertical or not.