PPL Assignment 6

Name: Himani Verma

Admission No: U19CS075

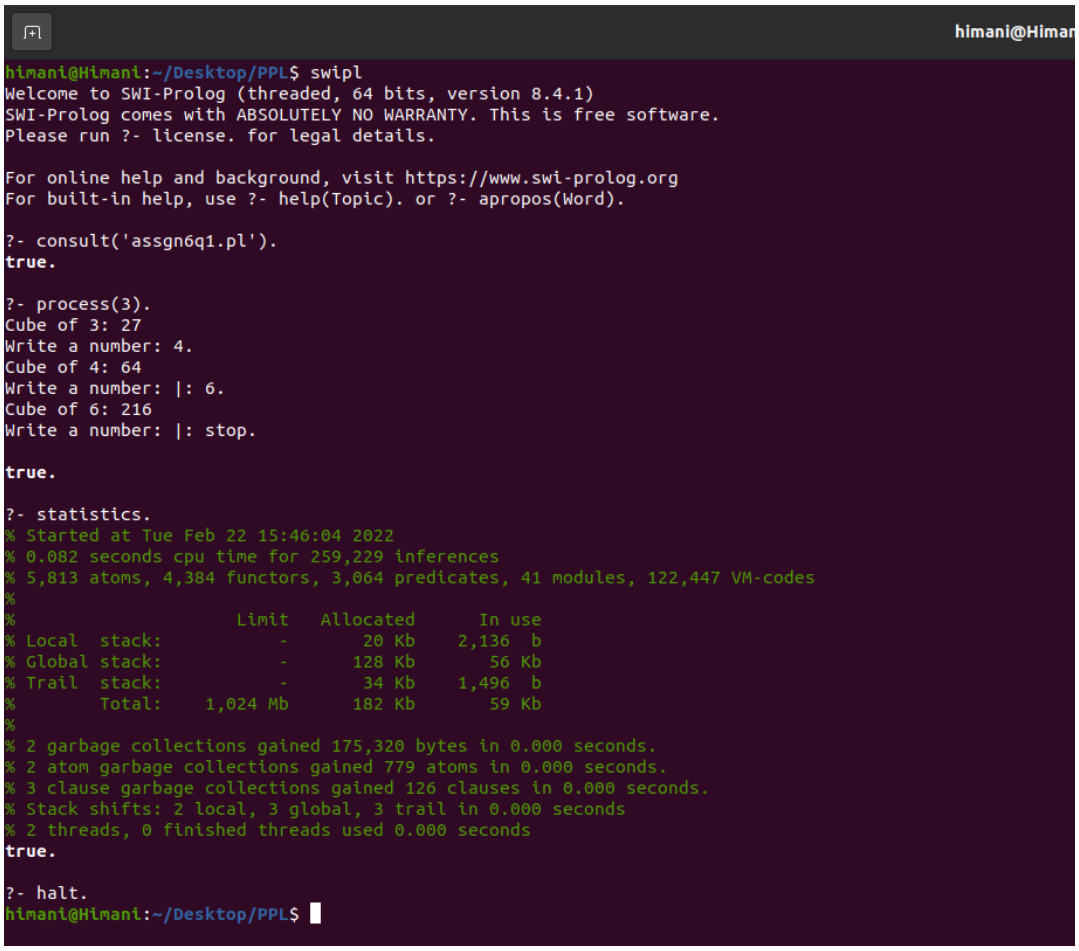
---

1. **Write a program in Prolog that uses following predicates**
   Write, nl, read, consult,halt,statistics.

   Source Code:

```
cube :-
write('Write a number: '),
read(Number),
process(Number).
process(stop) :- !.
process(Number) :-
C is Number * Number * Number,
write('Cube of '),write(Number),write(': '),write(C),nl, cube.
```

   Output:



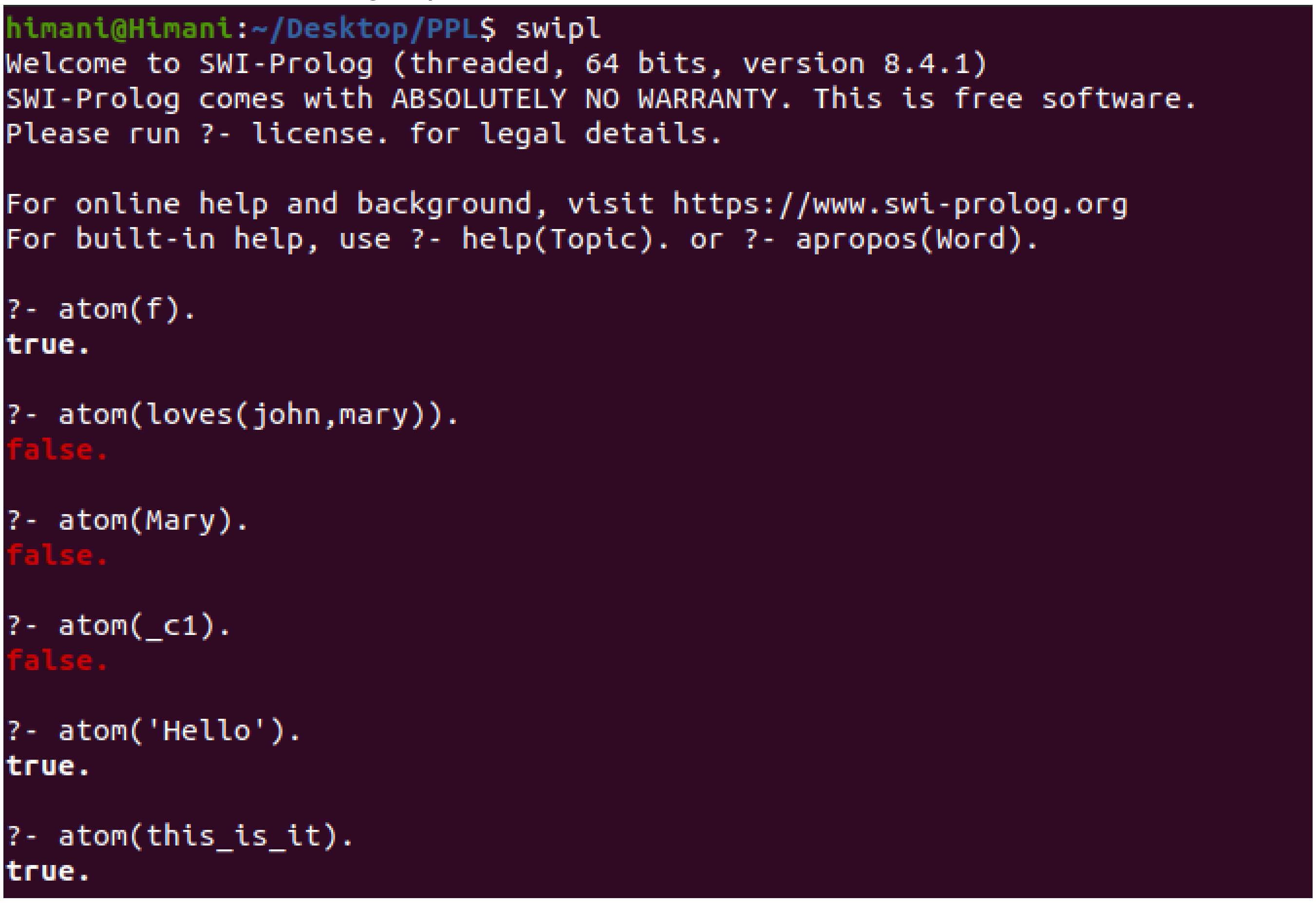
```
himani@Himani:~/Desktop/PPL$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- consult('assgn6q1.pl').
true.

?- process(3).
Cube of 3: 27
Write a number: 4.
Cube of 4: 64
Write a number: |: 6.
Cube of 6: 216
Write a number: |: stop.

true.

?- statistics.
% Started at Tue Feb 22 15:46:04 2022
% 0.082 seconds cpu time for 259,229 inferences
% 5,813 atoms, 4,384 functors, 3,064 predicates, 41 modules, 122,447 VM-codes
%
%                      Limit   Allocated      In use
% Local  stack:            -       20 Kb    2,136  b
% Global stack:            -      128 Kb       56 Kb
% Trail  stack:            -       34 Kb    1,496  b
%        Total:    1,024 Mb      182 Kb       59 Kb
%
% 2 garbage collections gained 175,320 bytes in 0.000 seconds.
% 2 atom garbage collections gained 779 atoms in 0.000 seconds.
% 3 clause garbage collections gained 126 clauses in 0.000 seconds.
% Stack shifts: 2 local, 3 global, 3 trail in 0.000 seconds
% 2 threads, 0 finished threads used 0.000 seconds
true.

?- halt.
himani@Himani:~/Desktop/PPL$ 
```

2. **Try to answer the following questions first "by hand" and then verify your answers using a Prolog interpreter.**
   (a) Which of the following are valid Prolog atoms?
   f, loves(john,mary), Mary, _c1, 'Hello', this_is_it
   (b) Which of the following are valid names for Prolog variables?
   a, A, Paul, 'Hello', a_123, _, _abc, x2
   (c) What would a Prolog interpreter reply given the following query?

?- f(a, b) = f(X, Y).
(d) Would the following query succeed?
?- loves(mary, john) = loves(John, Mary). Why?
(e) Assume a program consisting only of the fact a(B, B). has been consulted by Prolog. How will the system react to the following query? ?- a(1, X), a(X, Y), a(Y, Z), a(Z, 100). Why?

Solution:

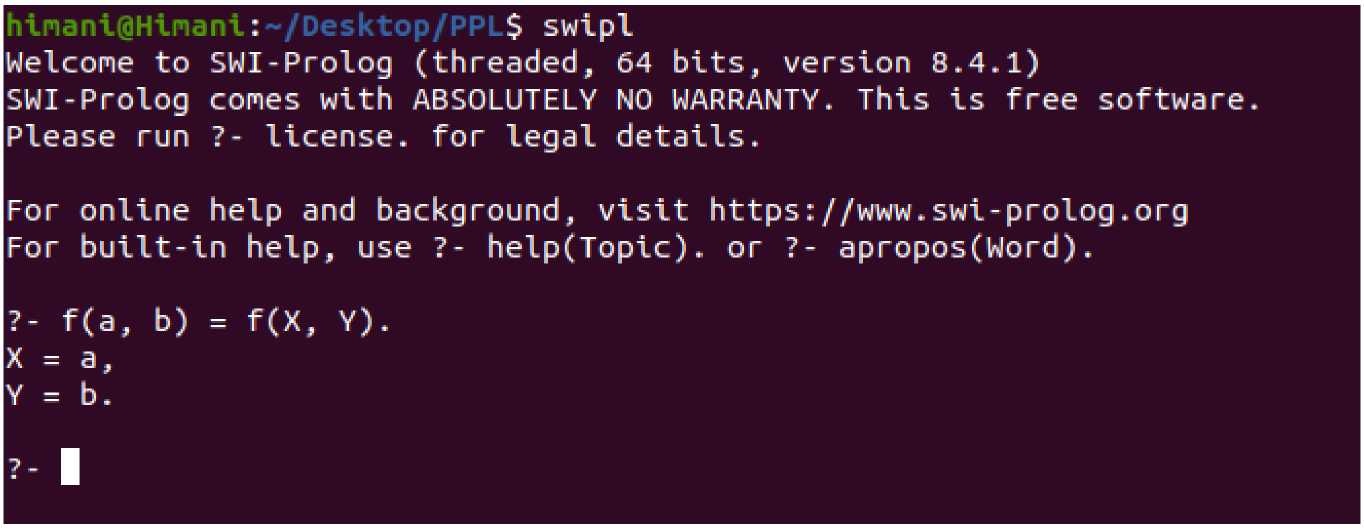
(a) Atoms are usually strings made up of lower- and uppercase letters, digits, and the underscore, starting with a lowercase letter. On top of that also any series of arbitrary characters enclosed in single quotes denotes an atom.



```
himani@Himani:~/Desktop/PPL$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- atom(f).
true.

?- atom(loves(john,mary)).
false.

?- atom(Mary).
false.

?- atom(_c1).
false.

?- atom('Hello').
true.

?- atom(this_is_it).
true.
```

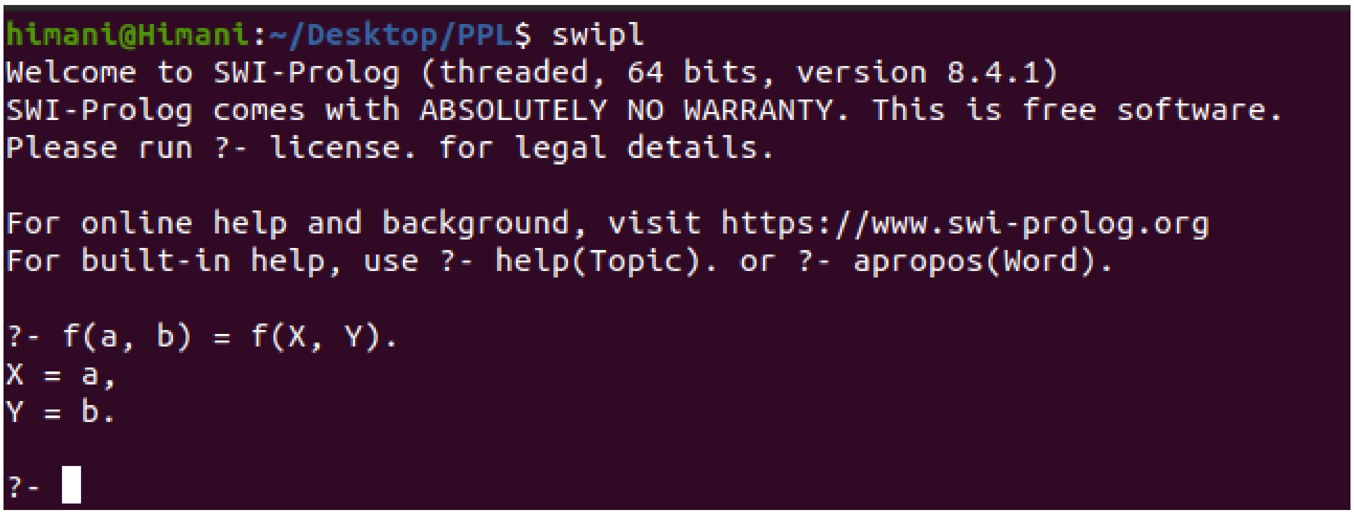
(b) A variable is a string of upper-case letters, lower-case letters, digits, and underscore characters that start either with an upper-case letter or with an underscore. A, Paul, _, _abc are valid variable names.

(c)



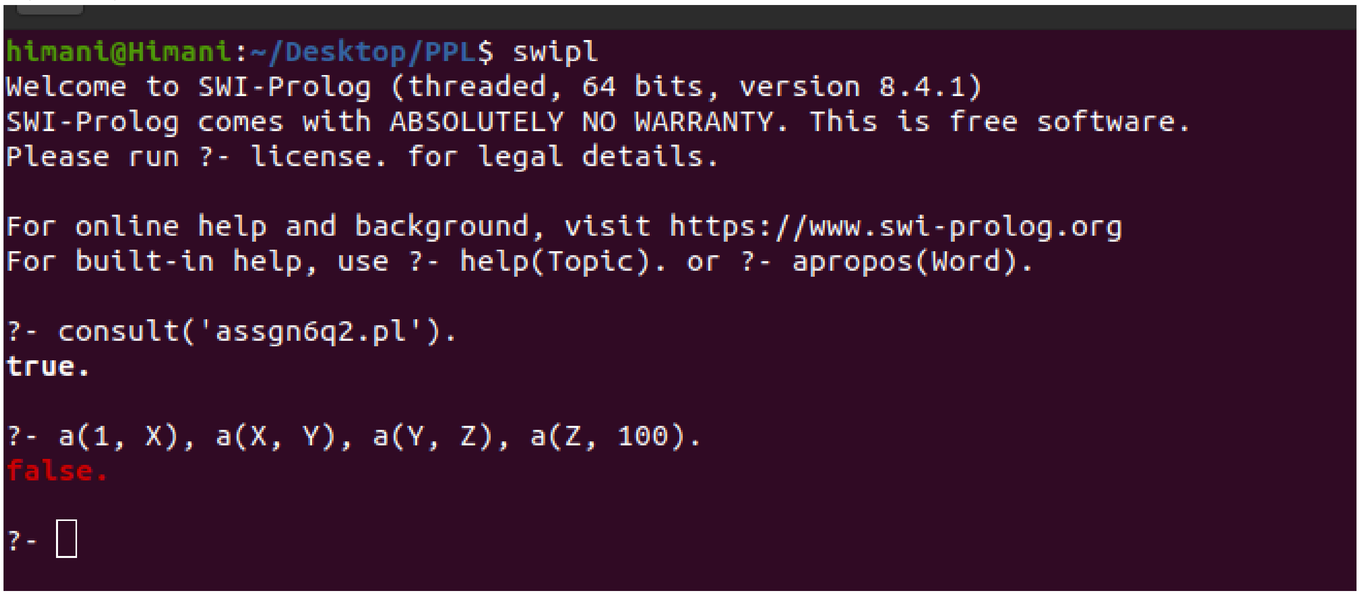
```
himani@Himani:~/Desktop/PPL$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- f(a, b) = f(X, Y).
X = a,
Y = b.

?- 
```

(d)
Yes, the query will be successfully executed as it takes mary and john as arguments while John and Mary of second predicate will be variables so it becomes the case of Q(C) which is mentioned above.



```
himani@Himani:~/Desktop/PPL$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- f(a, b) = f(X, Y).
X = a,
Y = b.

?-
```

(e)
The given function a(X ,Y) is explained as X=Y. Here, the definition of function a(X, Y) is a(B, B) implies both the parameter are equal which goves the following output.
a(1,X) => X = 1.
a(X,Y) => Y = X => Y = 1.
a(Y,Z) => Z = Y => Z = 1.
a(Z,100) => Z=100 => 1=100 return false.



```
himani@Himani:~/Desktop/PPL$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- consult('assgn6q2.pl').
true.

?- a(1, X), a(X, Y), a(Y, Z), a(Z, 100).
false.

?-
```

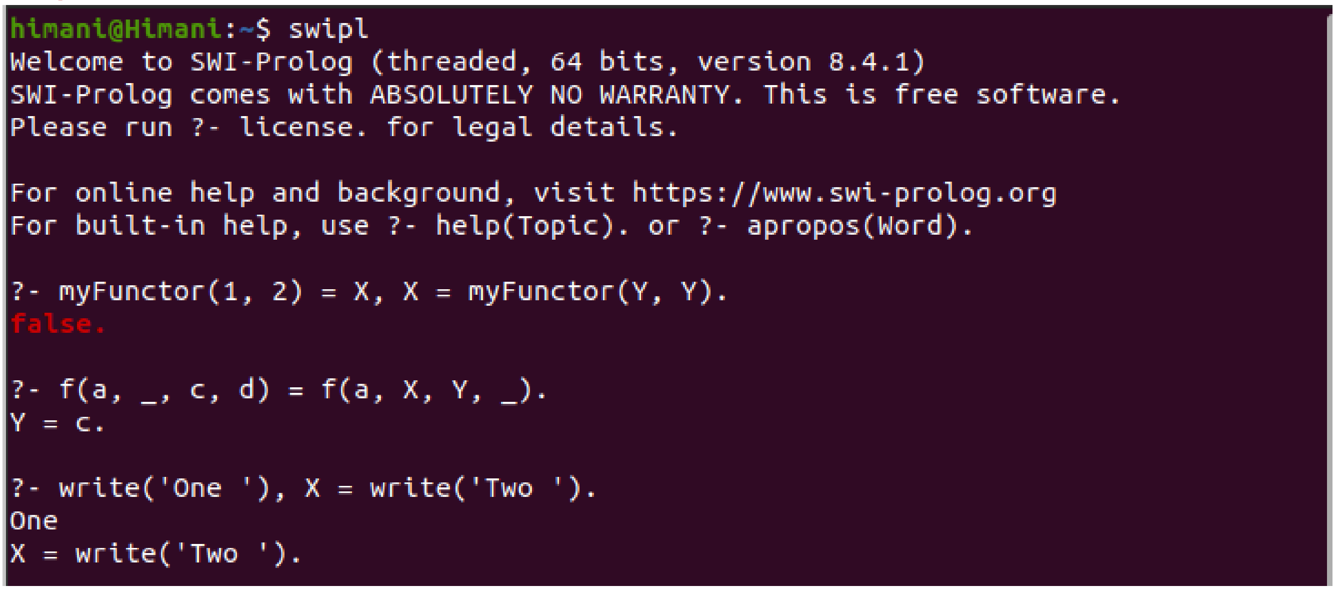
3. **Read the section on matching again and try to understand what's happening when you submit the following queries to Prolog.**
   (a) ?- myFunctor(1, 2) = X, X = myFunctor(Y, Y).
   (b) ?- f(a, _, c, d) = f(a, X, Y, _).
   (c) ?- write('One '), X = write('Two ').

   Source Code:

   In the **(a) part** the second predicate myFunctor(Y,Y) is defined when both the variables are same and it produces the output X which contradicts with the first predicate as both arguments differ in value and still provides with result X.

In **(b) part**, the value of Y can be equated to c while the value of X is not defined as it corresponds to _ (denotes an anonymous variable) in the above predicate.
In the **(c) part**, write('One') prints "One" on the console, and the second predicate assigns that value to the variable X.

Output:



```
himani@Himani:~$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- myFunctor(1, 2) = X, X = myFunctor(Y, Y).
false.

?- f(a, _, c, d) = f(a, X, Y, _).
Y = c.

?- write('One '), X = write('Two ').
One
X = write('Two ').
```

4. **Draw the family tree corresponding to the following Prolog**
   After having copied the program, define new predicates (in terms of rules using male/1, female/1 and parent/2) for the following family relations:
   (a) father
   (b) sister
   (c) grandmother
   (d) cousin
   You may want to use the operator \=, which is the opposite of =. A goal like X \= Y succeeds, if the two terms X and Y cannot be matched.

   Example: X is the brother of Y, if they have a parent Z in common and if X is male and if X and Y don't represent the same person. In Prolog this can be expressed through the following rule:

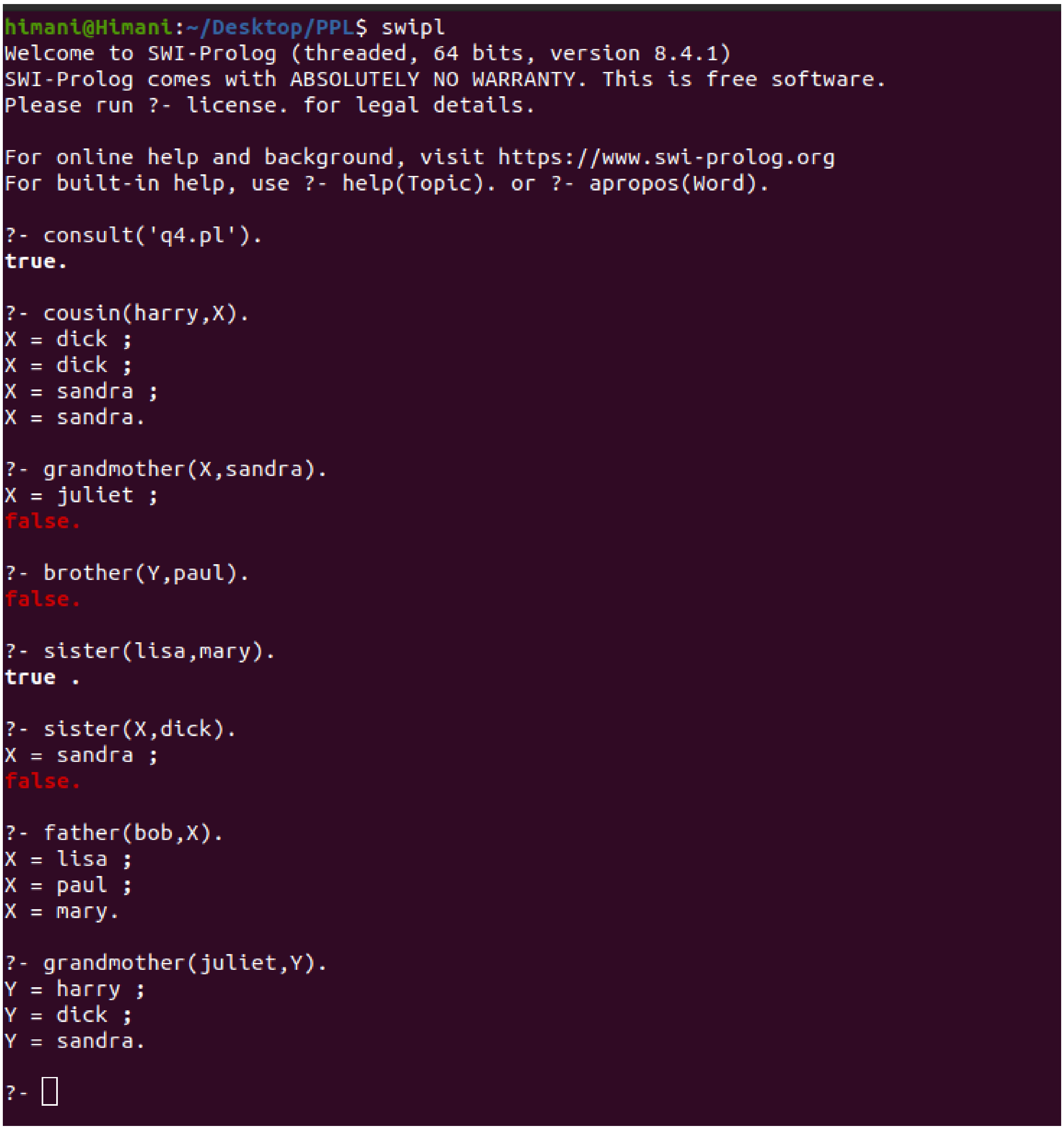
   brother(X, Y) :-
   parent(Z, X),
   parent(Z, Y),
   male(X),
   X \= Y.

Source Code:

```
%female
female(mary).
female(sandra).
female(juliet).
female(lisa).

%male
male(peter).
male(paul).
male(dick).
male(bob).
male(harry).
```

```
%parent
parent(bob, lisa).
parent(bob, paul).
parent(bob, mary).
parent(juliet, lisa).
parent(juliet, paul).
parent(juliet, mary).
parent(peter, harry).
parent(lisa, harry).
parent(mary, dick).
parent(mary, sandra).

%father
father(X,Y) :-
    male(X),
    parent(X,Y).

%sister
sister(X,Y) :-
    female(X),
    parent(Z,X),
    parent(Z,Y),
    X \= Y.

%brother
brother(X,Y) :-
    male(X),
    parent(Z,X),
    parent(Z,Y),
    X \= Y.

%grandmother
grandmother(X,Y) :-
    female(X),
    parent(X,Z),
    parent(Z,Y).

%cousin
cousin(X,Y) :-
    parent(Z,X),
    parent(W,Y),
    (brother(Z,W);sister(Z,W)).
```

Output:



```
himani@Himani:~/Desktop/PPL$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- consult('q4.pl').
true.

?- cousin(harry,X).
X = dick ;
X = dick ;
X = sandra ;
X = sandra.

?- grandmother(X,sandra).
X = juliet ;
false.

?- brother(Y,paul).
false.

?- sister(lisa,mary).
true .

?- sister(X,dick).
X = sandra ;
false.

?- father(bob,X).
X = lisa ;
X = paul ;
X = mary.

?- grandmother(juliet,Y).
Y = harry ;
Y = dick ;
Y = sandra.

?- 
```