CS2209 – Assignment 5

Pearson Radu Wed/Apr 6/16

Question 1

```
male(perry).
male(pearson).
male(dinis).
male(denis).
male(nathan).
male(anthony).

female(delainha).
female(isabel).
female(maria).
female(alana).

parent(perry, pearson).
parent(perry, delainha).
parent(isabel, pearson).
parent(isabel, delainha).
parent(dinis, isabel).
parent(dinis, denis).
parent(maria, isabel).
parent(maria, denis).
parent(denis, nathan).
parent(denis, anthony).
parent(alana, nathan).
parent(alana, anthony).

father(X,Y) :- parent(X,Y), male(X).
mother(Y,X) :- parent(Y,X), female(Y).
brother(X,Y) :- parent(Z,X), parent(Z,Y), male(X), X\=Y.
sibling(X,Y) :- parent(Z,X), parent(Z,Y), X\=Y.
grandson(X,Y) :- parent(Y,Z), parent(Z,X), male(X).
cousin(X,Y):-parent(A,X),     parent(B,Y), parent(W,A),     parent(W,B),   A\=B.
motherinlaw(X,Y) :- parent(X,W), parent(W,Z), parent(Y,Z), not(parent(X,Y)), female(X).
descendant(X,Y) :- parent(Y,X).
descendant(X,Y) :- parent(Y,Z), parent(Z,X).
```

SWI-Prolog -- e:/Documents/University/Year 2/Semester 2/CS2209/Assignments/Assignment 5/q...

File Edit Settings Run Debug Help

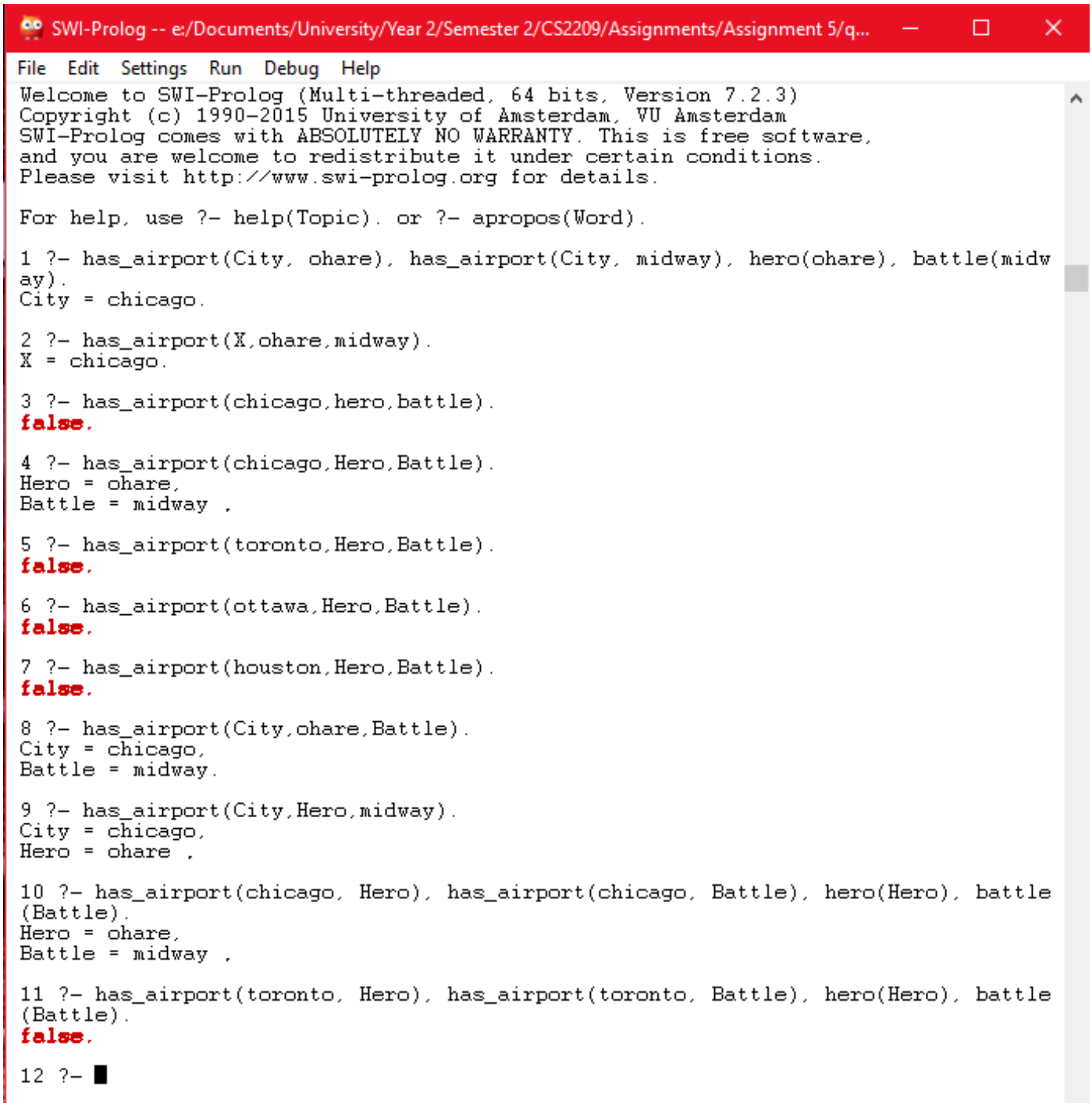
```
Welcome to SWI-Prolog (Multi-threaded, 64 bits, Version 7.2.3)
Copyright (c) 1990-2015 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org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

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

1 ?- father(dinis,X).
X = isabel ;
X = denis.

2 ?- mother(isabel,pearson).
true .

3 ?- brother(nathan,anthony).
true .

4 ?- sibling(pearson,delainha).
true .

5 ?- grandson(isabel,X).
false.

6 ?- grandson(X,dinis).
X = pearson ;
X = nathan ;
X = anthony.

7 ?- cousion(nathan,pearson).
Correct to: "cousin(nathan,pearson)"? yes
true .

8 ?- motherinlaw(maria,perry).
true .

9 ?- motherinlaw(dinis,perry).
false.

10 ?- descendant(pearson,perry).
true .

11 ?- descendant(perry,pearson).
false.

12 ?- descendant(pearson,X).
X = perry ;
X = isabel ;
X = dinis ;
X = maria ;
false.

13 ?-
```

Question 2

```
airport(pearson).
airport(ottawaInternational).
airport(vancouverInternational).
airport(ohare).
airport(newyorkInternational).
airport(losAngelesInternational).
airport(mccaran).

city(toronto).
city(ottawa).
city(vancouver).
city(chicago).
city(newyork).
city(losangeles).
city(lasvegas).

hero(ohare).
hero(churchill).
hero(eisenhower).
hero(patton).
hero(bader).
hero(reid).
hero(cheshire).

has_airport(toronto,pearson).
has_airport(ottawa,ottawaInternational).
has_airport(vancouver,vancouverInternational).
has_airport(chicago,ohare).
has_airport(newyork,newyorkInternational).
has_airport(losangeles,losAngelesInternational).
has_airport(lasvegas,mccaran).

battle(midway).
battle(atlantic).
battle(normandy).
battle(dunkirk).
battle(astonia).
battle(cherbourg).
battle(lille).

has_airport(toronto,bishop).
has_airport(ottawa,rockcliffe).
has_airport(vancouver,langley).
has_airport(chicago,midway).
has_airport(newyork,newark).
```

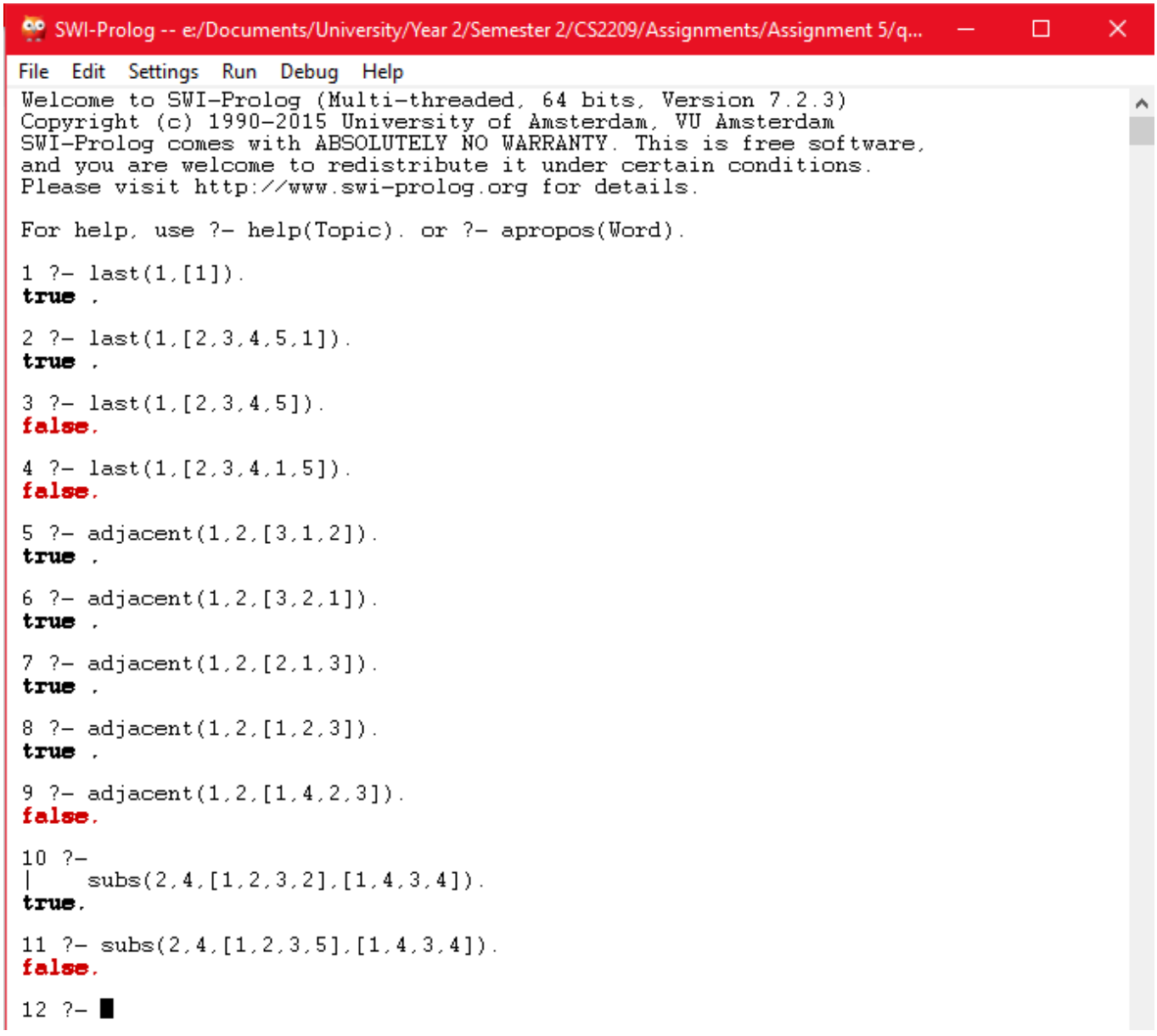
has_airport(losangeles,wayne).
has_airport(lasvegas,north).

has_airport(X,Y,Z) :- has_airport(X,Y),has_airport(X,Z),hero(Y),battle(Z),Y\=Z.



Question 3

```
last(X,[X]).
last(X,[_|Xs]) :- last(X,Xs).

adjacent(X,Y,[X,Y,|_]).
adjacent(X,Y,[_|Zs]) :- adjacent(X,Y,Zs).

subs(_, _, [], []).
subs(X, Y, [X|T1], [Y|T2]) :- subs(X, Y, T1, T2), !.
subs(X, Y, [H|T1], [H|T2]) :- \+ is_list(H), subs(X, Y, T1, T2), !.
subs(X, Y, [H1|T1], [H2|T2]) :- subs(X, Y, H1, H2), subs(X, Y, T1, T2).
```



Question 4

```
x(N).
o(N).

ordered_line(1,2,3).
ordered_line(4,5,6).
ordered_line(7,8,9).
ordered_line(1,4,7).
ordered_line(2,5,8).
ordered_line(3,6,9).
ordered_line(1,5,9).
ordered_line(3,5,7).
line(A,B,C) :- ordered_line(A,B,C).
line(A,B,C) :- ordered_line(A,C,B).
line(A,B,C) :- ordered_line(B,A,C).
line(A,B,C) :- ordered_line(B,C,A).
line(A,B,C) :- ordered_line(C,A,B).
line(A,B,C) :- ordered_line(C,B,A).

move(A) :- good(A), empty(A).
full(A) :- x(A).
full(A) :- o(A).
empty(A) :- not(full(A)).
good(A) :- win(A).
good(A) :- block_win(A).
good(A) :- spilt(A).
good(A) :- block_split(A).
good(A) :- build(A).

win(A) :- x(B),x(C),line(A,B,C).
block_win(A) :- o(B),o(C),line(A,B,C).
split(A) :- x(B),x(C),different(B,C),line(A,B,D),line(A,C,E),empty(D),empty(E).
same(A,A).
different(A,B) :- not(same(A,B)).
block_split(A) :- o(B),o(C),different(B,C),line(A,B,D),line(A,C,E),empty(D),empty(E).
build(A) :- x(B),line(A,B,C),empty(C).

good(5).
good(1).
good(3).
good(7).
good(9).
good(2).
good(4).
good(6).
good(8).
```

Question 5

```
word(dog,d,o,g).
word(run,r,u,n).
word(top,t,o,p).
word(five,f,i,v,e).
word(four,f,o,u,r).
word(lost,l,o,s,t).
word(mess,m,e,s,s).
word(unit,u,n,i,t).
word(bake,b,a,k,e).
word(forum).
word(f,o,r,u,m).
word(super,s,u,p,e,r).
word(prolog,p,r,o,l,o,g).
word(wonder,w,o,n,d,e,r).
word(vanish,v,a,n,i,s,h).
word(yellow,y,e,l,l,o,w).
```