Programming Languages Project 3 - Robert Donohue

Full prolog code as text:

```
%flight(number, start, dest, departuretime, arrivaltime)

flight(6711, bos, ord, 0815, 1005).
flight(211, lga, ord, 0700, 0830).
flight(203, lga, lax, 0730, 1335).
flight(92221, ewr, ord, 0800, 0920).
flight(2134, ord, sfo, 0930, 1345).
flight(954, phx, dfw, 1655, 1800).
flight(1176, sfo, lax, 1430, 1545).
flight(205, lax, lga, 1630, 2210).
flight(7791, lga, ord, 0815, 0945).
flight(321, lax, lga, 1645, 2225).

%rule1 - destination from PHX:
destination_from_phx(Destination) :- flight(_,phx,Destination,_,_).

%rule2 - flight to PHX?:
flight_to_phx(Flight) :- flight(Flight,_,phx,_,_).

%rule3 - bos to ord landing times
bos_to_ord_time(Time) :- flight(_, bos, ord, _, Time).

%rule4 - does ord to sfo depart after ewr to ord
ord_sfo_after_ewr_ord(Flight1, Flight2) :-
        flight(Flight1, ord, sfo, Dep, _),
        flight(Flight2, ewr, ord, _, Ariv),
        Dep > Ariv.

%rule5 - ord arrival times
ord_arrival_time(Time) :- flight(_, _, ord, _, Time).


%rule6 - what are all the ways to get from lga to lax
%(respect that next departure must be before previous arrival for connections)
%write a general 'path' function that determines if a path exists between two airports
%keeps flight numbers in a list and appends visited airports to a list to avoid loops

path([FlightNumbers], Start, Dest, Visited) :-
        flight(FlightNumbers, Start, Dest, _, _),
        \+ member(Dest, Visited). %member checks if Dest is already in the Visited list
```

```
%recursive function that relies on path function, checks connections:
path([FlightNumbers | Remainingflights], Start, Dest, Visited) :-
        flight(FlightNumbers, Start, Mid, _, Arrivaltime),
        Mid \= Dest, %if mid isn't the Destination
        \+ member(Mid, Visited), %if mid isn't in the visited list

        %recursive call
        path(Remainingflights, Mid, Dest, [Mid | Visited]),
        %check if layover times make sense:
        flight(NextflightNum, Mid, _, NextDepartureTime, _),
        member(NextflightNum, Remainingflights), %flight needs to be in the remaining list
        Arrivaltime < NextDepartureTime. %ensure that the arrival time is before your next
departure


%make path recursive function specific to lga to lax:

lga_to_lax(Flights) :-
        path(Flights,lga, lax, [lga]). %visited starts with lga
```

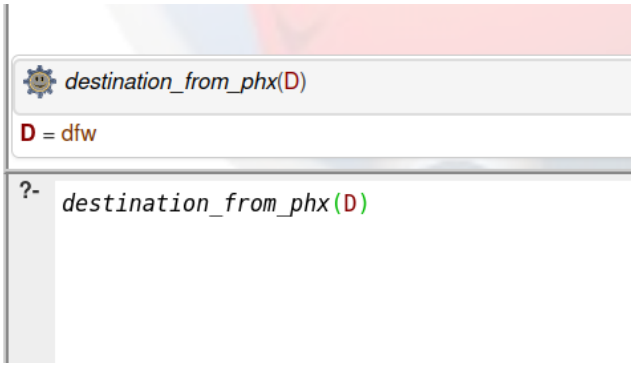
Full code as screenshots:

```
2
3  %flight(number, start, dest, departuretime, arrivaltime)
4
5  flight(6711, bos, ord, 0815, 1005).
6  flight(211, lga, ord, 0700, 0830).
7  flight(203, lga, lax, 0730, 1335).
8  flight(92221, ewr, ord, 0800, 0920).
9  flight(2134, ord, sfo, 0930, 1345).
10 flight(954, phx, dfw, 1655, 1800).
11 flight(1176, sfo, lax, 1430, 1545).
12 flight(205, lax, lga, 1630, 2210).
13 flight(7791, lga, ord, 0815, 0945).
14 flight(321, lax, lga, 1645, 2225).
15
16 %rule1 - destination from PHX:
17 destination_from_phx(Destination) :- flight(_,phx,Destination,_,_).
18
19 %rule2 - flight to PHX?:
20 flight_to_phx(Flight) :- flight(Flight,_,phx,_,_).
21
22 %rule3 - bos to ord landing times
23 bos_to_ord_time(Time) :- flight(_, bos, ord, _, Time).
24
25 %rule4 - does ord to sfo depart after ewr to ord
26 ord_sfo_after_ewr_ord(Flight1, Flight2) :-
27     flight(Flight1, ord, sfo, Dep, _),
28     flight(Flight2, ewr, ord, _, Ariv),
29     Dep > Ariv.
30
31 %rule5 - ord arrival times
32 ord_arrival_time(Time) :- flight(_, _, ord, _, Time).
33
34
35 %rule6 - what are all the ways to get from lga to lax
```

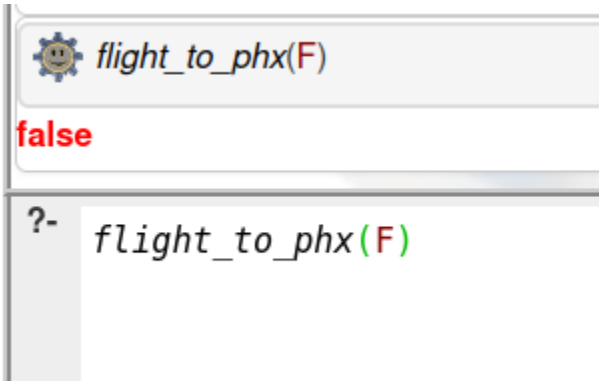
```
34
35 %rule6 - what are all the ways to get from lga to lax
36 %(respect that next departure must be before previous arrival for connections)
37 %write a general 'path' function that determines if a path exists between two airports
38 %keeps flight numbers in a list and appends visited airports to a list to avoid loops
39
40 path([FlightNumbers], Start, Dest, Visited) :-
41     flight(FlightNumbers, Start, Dest, _, _),
42     \+ member(Dest, Visited). %member checks if Dest is already in the Visited list
43
44
45 %recursive function that relies on path function, checks connections:
46 path([FlightNumbers | Remainingflights], Start, Dest, Visited) :-
47     flight(FlightNumbers, Start, Mid, _, Arrivaltime),
48     Mid \= Dest, %if mid isn't the Destination
49     \+ member(Mid, Visited), %if mid isn't in the visited list
50
51     %recursive call
52     path(Remainingflights, Mid, Dest, [Mid | Visited]),
53     %check if layover times make sense:
54     flight(NextflightNum, Mid, _, NextDeparturetime, _),
55     member(NextflightNum, Remainingflights), %flight needs to be in the remaining list
56     Arrivaltime < NextDeparturetime. %ensure that the arrival time is before your next departure
57
58
59 %make path recursive function specific to lga to lax:
60
61 lga_to_lax(Flights) :-
62     path(Flights,lga, lax, [lga]). %visited starts with lga
63
64
```

Queries:

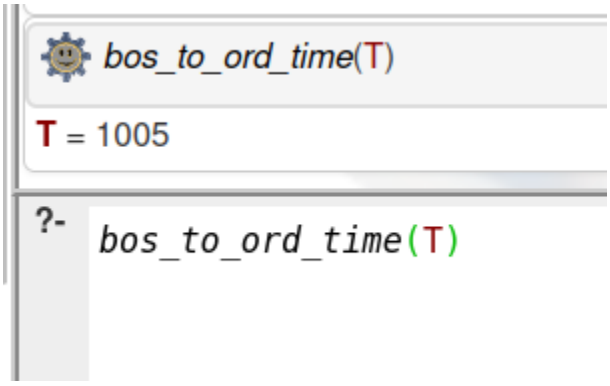
1) destination_from_phx(D)



2) flight_to_phx(F)



3) bos_to_ord_time(T)



4) ord_sfo_after_ewr_ord(F1, F2)

ord_sfo_after_ewr_ord(F1,F2)

F1 = 2134,
F2 = 92221

```
?- ord_sfo_after_ewr_ord(F1,F2)
```

5) ord_arrival_time(T)

ord_arrival_time(T)

T = 1005
T = 830
T = 920
T = 945

```
?- ord_arrival_time(T)
```

6) lga_to_lax(Flights)

lga_to_lax(F)

F = [203]
F = [211, 2134, 1176]
false

```
?- lga_to_lax(F)
```